VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra elektroniky

Řízení výstupního napětí nepřímého měniče kmitočtu s napěťovým meziobvodem

Output Voltage Control of Frequency Converter with Voltage Source Inverter

2016 Bc. Marián Ruský

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra elektroniky

Zadání diplomové práce

Student: **Bc. Marián Ruský**

Studijní program: N2649 Elektrotechnika

Studijní obor: 2612T015 Elektronika

Téma: Řízení výstupního napětí nepřímého měniče frekvence s napěťovým meziobvodem
Output Voltage Control of Frequency Converter with Voltage Source Inverter

Jazyk vypracování: čeština

Zásady pro vypracování:

1. Proveďte teoretický rozbor možností řízení výstupního napětí trojfázového napěťového střídače.
2. Do řídicí jednotky s DSP TMS320F28335 implementujte vybrané algoritmy řízení výstupního napětí trojfázového napěťového střídače.
3. Experimentálně ověřte správnou činnost realizovaných algoritmů na laboratorním stanovišti s asynchronním motorem a proveďte měření nejdůležitějších veličin.

Seznam doporučené odborné literatury:

Dle pokynů vedoucího závěrečné práce

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Martin Kuchař, Ph.D.**

Datum zadání: 01.09.2015

Datum odevzdání: 29.04.2016

doc. Ing. Petr Palacký, Ph.D.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

VYSOKÁ ŠKOLA BÁŇSKÁ - TECHNICKÁ UNIVERZITA * Fakulta elektrotechniky a informatiky * OSTRAVA *

Prohlášení studenta

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.
Dne: 22.4.2014

.....

Podpis

Poděkování

Rád bych poděkoval vedoucímu diplomové práce **Ing. Martin Kuchař, Ph.D.** za odbornou pomoc a konzultaci při vytváření této práce.

Abstrakt

Cílem této diplomové práce je implementace tří vybraných algoritmů řízení výstupního napětí nepřímého měniče kmitočtu s napěťovým meziobvodem do řídicího systému s DSP TMS320F28335, který umožňuje operace s plovoucí řádovou čárkou. V práci jsou vysvětleny principy vybraných metod řízení a dále se práce se zabývá především samotným řešením aplikačního software procesoru a nastavením jeho periferií tak aby plnily požadovanou funkci. Pro potřeby komunikace mezi uživatelem a řídicím systémem je sestaveno jednoduché uživatelské rozhraní v prostředí Labview. Realizovaný aplikační software je na konci práce ověřen při řízení zmíněného měniče napájejícího asynchronní motor.

Klíčová slova

Metody řízení nepřímého měniče kmitočtu, komparační PWM, vektorová PWM, digitální signálový procesor, Labview.

Abstract

Target of this diploma thesis is implementation of three chosen algorithms of output voltage control of frequency converter with voltage source inverter into a control system with DSP TMS320F28335, which allows operations with floating point. Principles of chosen methods of controlling the output voltage are explained in the thesis and above all the thesis deals with implementation of these methods into the application software of the processor and settings of its peripheries so they fulfill the desired function. A simple user interface is built in Labview for the need of communication between the control system and user. Implemented application software is finally verified by controlling the said frequency converter driving an asynchronous motor.

Key words

Methods of controlling an output voltage of frequency converter, comparative PWM, space vector PWM, digital signal processor, Labview

Seznam použitých symbolů a zkratek

PWM – pulsně šířková modulace (Pulse Width Modulation)
DSP – Digitální Signálový Procesor
NMK – Nepřímý Měnič Kmitočtu
ePWM – Enhanced Pulse Width Modulator
f_{ref} – frekvence referenčního sinusového signálu
f_{out} – výstupní frekvence
f_{sw} – spínací/vzorkovací frekvence
T_{sw} – spínací/vzorkovací perioda
T_1 – doba sepnutí prvního aktivního napěťového vektoru
T_2 – doba sepnutí druhého aktivního napěťového vektoru
T_0 – doba sepnutí nulového napěťového vektoru
U_{ref} – amplituda referenčního sinusového signálu
U_{out} – amplituda výstupního fázového napětí NMK
U_T – amplituda pilovitého signálu
U_d – napětí meziobvodu
U_{RMS} – efektivní hodnota napětí
U_{skut} – skutečné napětí meziobvodu
$U_{měř}$ – napětí meziobvodu měřené DSP
$u(t)_{ref\,a,b,c}$ – časový průběh referenčních sinusových signálů
u_α* – časová průběh α složky vektoru napětí v systému statorových souřadnic
u_β* – časová průběh β složky vektoru napětí v systému statorových souřadnic
ω_{ref} – úhel pro výpočet funkce sinus při tvorbě referenčního signálu
m – hloubka modulace
n_k – neuron 1 až 6
n_i – hodnota vybraného neuronu s nejvyšším výsledkem
n_{i+1} – hodnota vybraného neuronu s druhým nejvyšším výsledkem
t_i – čas sepnutí aktivního napěťového vektoru, který byl vybrán na základě výsledku n_i
t_{i+1} – čas sepnutí vektoru aktivního napěťového, který byl vybrán na základě výsledku n_{i+1}
$\underline{\mathbf{u}}$* – výsledný vektor výstupního napětí
$\underline{\mathbf{u}}_0$ – $\underline{\mathbf{u}}_7$ – napěťové vektory realizované kombinací sepnutí výkonových spínačů

Obsah

1 Úvod ... 1
2 Teorie řízení výstupního napětí NMK ... 2
2.1 Komparační PWM ... 3
2.2 Komparační PWM s přidáním třetí harmonické ... 3
2.3 Vektorová PWM s využitím kompetitivní neuronové sítě ... 4
3 Popis důležitých parametrů použitého hardwaru ... 8
3.1 Budiče výkonových prvků řízeného NMK ... 8
3.2 DSP TMS320F28335 ... 8
3.3 ePWM jednotka DSP TMS320F28335 ... 9
3.3.1 Časová základna TB ... 10
3.3.2 Komparační jednotka CC ... 10
3.3.3 Vyhodnocovací jednotka AQ ... 10
3.3.4 Spouštěč událostí ET ... 11
3.3.5 Generátor bezpečností doby DB ... 11
4 Popis nastavení DSP TMS320F28335 ... 13
4.1 Časování přerušení ... 13
4.2 Nastavení jednotek ePWM ... 15
5 Popis aplikačního softwaru pro DSP ... 18
5.1 Vývojový diagram a popis činnosti aplikačního software ... 19
5.2 Realizace komparační PWM ... 23
5.3 Realizace komparační PWM s přidáním třetí harmonické ... 24
5.4 Realizace vektorová PWM s využitím kompetitivní neuronové sítě ... 24
6 Uživatelské rozhraní ... 28
6.1 Komunikace s řídicím systémem ... 29
7 Experimentální výsledky na laboratorním stanovišti ... 32
7.1 Komparační PWM ... 32
7.2 Komparační PWM s přidáním třetí harmonické ... 36
7.3 Vektorová PWM s využitím kompetitivní neuronové sítě ... 40
7.4 Zhodnocení získaných výsledků ... 44
8 Závěr ... 45

1 Úvod

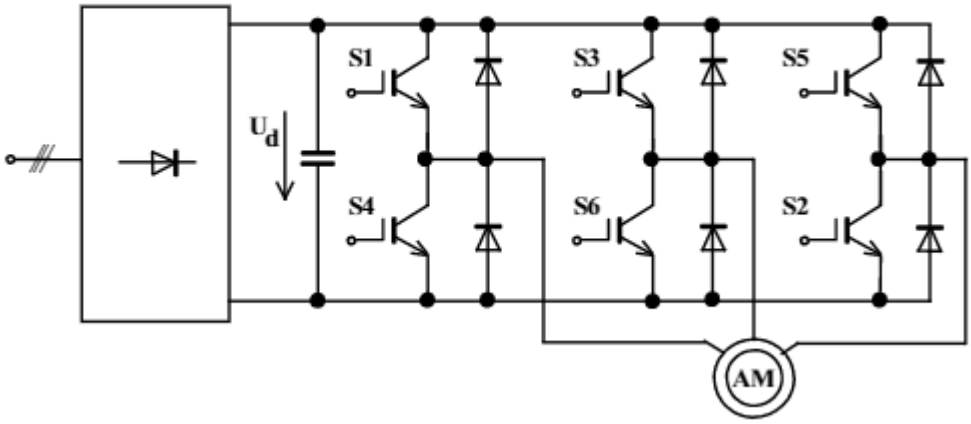
V současnosti je jak v komerčních, tak i v průmyslových aplikacích stále rozšířenější použití asynchronních motorů. Asynchronní motory postupně vytlačují pohony se stejnosměrnými motory především díky jejich jednoduché konstrukci, která je činí téměř bezúdržbovými, což se o stejnosměrných motorech opatřených mechanickým komutátorem říct nedá. Asynchronní motory jsou díky absenci komutátoru a s ním spojeným jiskřením, které je rovněž zdrojem elektromagnetického rušení, vhodné pro použití v nebezpečných a výbušných prostředích.

Pohonné systémy obsahující asynchronní motory jsou obvykle vybaveny měniči napětí, které jsou v závislosti na požadavcích dané aplikace řízeny celou řadou algoritmů. Řízení asynchronních motorů v dynamicky nenáročných aplikacích je řešeno nejčastěji jako skalární řízení. K tomuto typu řízení asynchronních motorů je třeba mít možnost měnit parametry napětí napájejícího daný motor, kterými jsou efektivní hodnota napětí a jeho kmitočet. Díky pokročilé výpočetní technice a stále výkonnějším signálovým procesorům, které dnes na trhu nabízí mnoho výrobců v celé řadě provedení, jsou v současnosti měniče napětí téměř výhradně řízeny systémy osazenými digitálními signálovými procesory specializovanými přímo pro řízení měničů napětí.

Cílem této práce je implementace tří vybraných metod řízení výstupního napětí NMK s napěťovým meziobvodem do řídicího systému osazeného digitálním signálovým procesorem. V následujících kapitolách tedy budou podrobněji rozebrány vybrané metody řízení výstupního napětí NMK, vlastnosti a technické specifikace periferií použitého DSP a implementace těchto algoritmů do řídicího systému s DSP TMS320F28335, který je vybaven periferiemi vhodnými právě pro řízení měničů napětí. Obsahem této práce je rovněž návrh uživatelského prostředí, kterým budou zadávány žádané parametry výstupního napětí a na závěr jsou vybrané metody a jejich funkčnost demonstrovány měřením na laboratorním stanovišti s NMK napájejícím asynchronní motor.

2 Teorie řízení výstupního napětí NMK

Nepřímé měniče kmitočtu s napěťovým meziobvodem jsou tvořeny zpravidla neřízeným šesti-pulzním usměrňovačem na vstupní straně, kondenzátory jakožto zdrojem vyhlazeného stejnosměrného napětí a trojfázovým střídačem v můstkovém zapojení na straně výstupní. K buzení spínacích prvků střídače jsou využívány obvody určené pro vhodnou úpravu řídícího signálu na budící signál pro spínací prvky měniče - tzv. budiče. K budičům je přiváděno řídící napětí z řídicího systému měniče. Zjednodušené zapojení výkonové části takového měniče je uvedeno na obrázku 1.



Obr. 1 - Zjednodušené zapojení NMK s napěťovým meziobvodem [1]

K řízení výstupní frekvence napětí a jeho amplitudy se používá pulzně šířková modulace výstupního napětí, čímž je možné měnit jeho efektivní hodnotu. Ať se jedná o jakoukoliv metodu řízení, může být v každé větvi sepnut vždy jen jeden ze dvojice spínačů, což umožňuje vytvořit celkem 6 různých kombinací sepnutí a navíc k nim 2 kombinace nazývané nulové stavy. Nulový stav je vytvořen sepnutím buďto všech horních nebo všech dolních výkonových spínačů (S1,S3,S5 nebo S4,S6,S2). Sepnutím takovéto kombinace se na statoru motoru vytvoří nulové napětí a energie nashromážděná v magnetickém obvodu motoru se ve formě elektrického proudu přes výkonové spínače uzavírá, což má za následek zastavení vektoru točivého magnetického pole.

K účelu řízení výstupního napětí NMK se využívá několik typů řídících algoritmů, jejichž implementace a rozbor jsou náplní následujících kapitol. Mezi tyto algoritmy patří např. komparační pulzně šířková modulace nebo vektorová pulzně šířková modulace.

Implementovány budou tři vybrané metody řízení výstupního napětí NMK. Před rozebráním jejich samotné implementace je nutno objasnit principy funkce těchto metod, což je cílem této kapitoly.

Zvoleny byly následující tři metody řízení výstupního napětí NMK

- Komparační pulzně šířková modulace
- Komparační pulzně šířková modulace s přidáním třetí harmonické
- Vektorová pulzně šířková modulace s využitím kompetitivní neuronové sítě

2.1 Komparační PWM

Komparační pulzně šířková modulace je tvořena porovnáváním tří referenčních sinusových signálů vzájemně posunutých o 120° o frekvenci f_{ref} odpovídající žádané výstupní frekvenci f_{out} a aplitudou U_{ref} s pilovitým průběhem o frekvenci f_{sw} a amplitudou U_T. Kdykoliv dojde k rovnosti referenčního signálu s pilovitým, je generován pulz pro sepnutí jednoho z dvojice spínačů v dané větvi. V opačném případě – tedy v době do rovnosti dvou zmíněných signálů, je sepnut druhý z dvojice spínačů v dané větvi. Z uvedeného je tedy zřejmé, že frekvence pilovitého signálu vlastně určuje frekvenci spínání výkonových spínačů střídače.

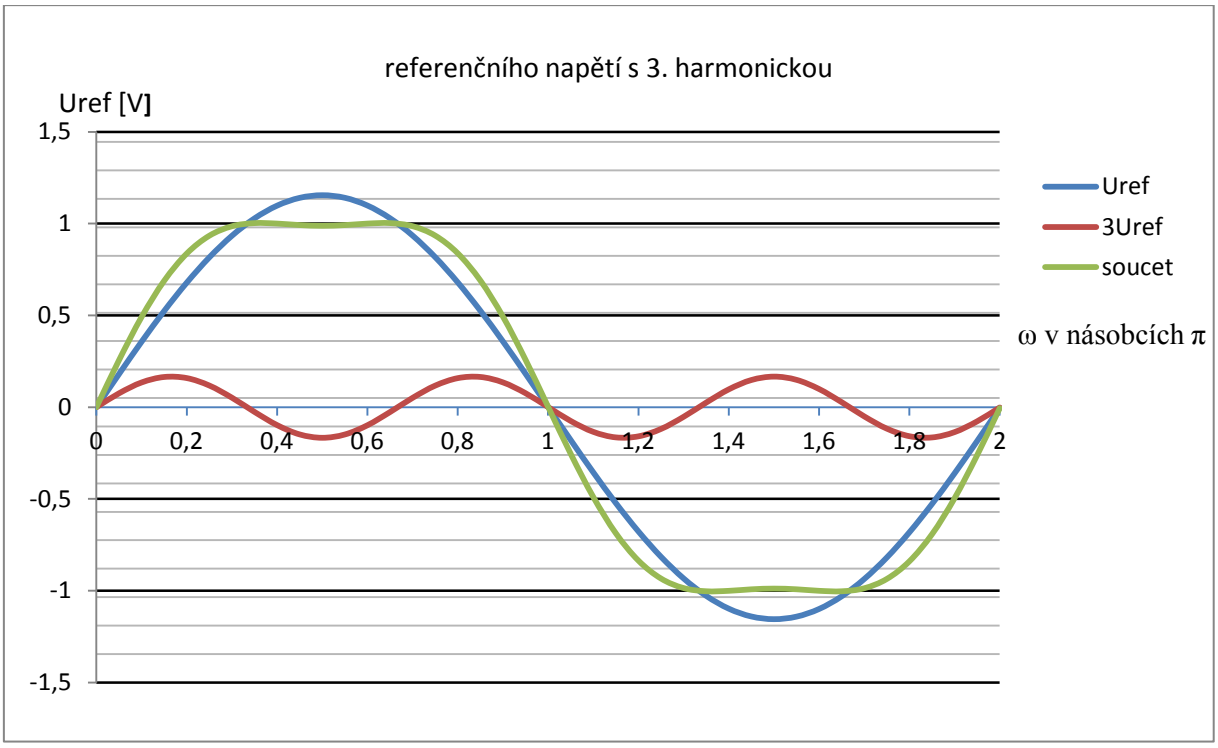
To jaký je poměr mezi amplitudou sinusových signálů U_{ref} a pilovitého signálu U_T se nazývá hloubkou modulace m. Hloubce modulace tedy odpovídá amplituda výstupního napětí a platí pro ni vztah 1. Konstanta 0,5, kterou je ve vztahu 1 násobeno napětí U_d je dána tím, že při využití této metody nelze napětí meziobvodu více využít [1]. Po vypočtení hloubky modulace v závislosti na požadované amplitudě výstupního napětí a napětí U_d, které je přítomno v meziobvodu, je pak možné vypočíst referenční sinusový signál dle vztahu 2.

$$m = \frac{U_{ref}}{U_T} \Rightarrow \frac{Uout}{0{,}5 * U_d} \tag{1}$$

$$u(t)_{ref\ a,b,c} = m * \sin(\omega_{ref} t + 0, +\frac{2}{3}\pi, -\frac{2}{3}\pi) \tag{2}$$

2.2 Komparační PWM s přidáním třetí harmonické

Hlavní výhodou přidání třetí harmonické do referenčního signálu je o 15% lepší využití napětí meziobvodu. Maximální dosažitelná amplituda výstupního signálu tedy není $0{,}5U_d$ jako v případě běžné komparační PWM, ale 0,577 U_d, což je znatelný rozdíl. Třetí harmonická sinusového referenčního signálu má amplitudu $\frac{1}{6}$ U_{ref}. Je nutno si uvědomit, že součtem těchto dvou signálů klesne maximální amplituda výsledného signálu, z tohoto důvodu je nutno první harmonickou referenčního signálu násobit koeficientem 1,155 [2], potom je při maximální hloubce modulace dosažena stejná amplituda výsledného signálu, jako u přechozí metody. Konstrukce referenčního signálu s přidáním třetí harmonické je k vidění na obrázku 2.



Obr. 2 - Referenční napětí s přidáním třetí harmonické

Jak vyplývá z výše uvedeného, změní se vztah pro výpočet požadované hloubky modulace v závislosti na zadané amplitudě výstupního napětí. Ze vztahu 3 je zřejmé, že při stejné hloubce modulace bude dosaženo vyššího výstupního napětí. Vztah 4 pak uvádí výpočet výsledného referenčního signálu.

$$m = \frac{U_{ref}}{U_T} \Rightarrow \frac{Uout}{0{,}575 * U_d} \tag{3}$$

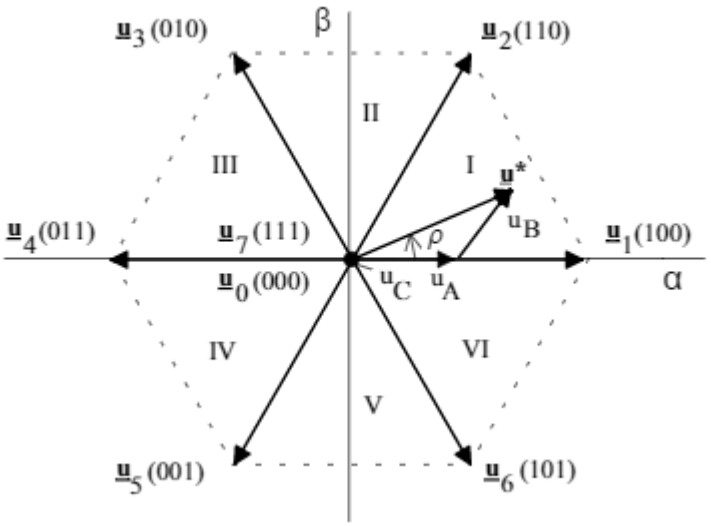
$$u(t)_{ref\ a,b,c} = m * \sin\left(\omega_{ref} t + 0, +\frac{2}{3}\pi, -\frac{2}{3}\pi\right) + \frac{1}{6}\sin(3\omega_{ref} t) \tag{4}$$

2.3 Vektorová PWM s využitím kompetitivní neuronové sítě

V případě vektorové pulzně šířkové modulace je vypočítávána požadovaná poloha a velikost prostorového vektoru výstupního napětí $\underline{\mathbf{u}}^*$. Poloha tohoto vektoru je pak realizována příslušnou kombinací sepnutí výkonových spínačů, jeho velikost pak dobou sepnutí těchto spínačů. Jak již bylo zmíněno v úvodu, můstkový trojfázový střídač je schopen realizovat 6 napěťových vektorů $\underline{\mathbf{u}}_1$ až $\underline{\mathbf{u}}_6$ a dva nulové vektory $\underline{\mathbf{u}}_0$ a $\underline{\mathbf{u}}_7$. Oblast, ve které se může vektor $\underline{\mathbf{u}}^*$ pohybovat je rozdělena na 6 sektorů.

Abychom byly schopni polohu vektoru měnit plynule, je třeba prokládat vždy dva vektory v příslušném sektoru, které jsou přilehlé k požadovanému vektoru výstupního napětí $\underline{\mathbf{u}}^*$. Poměrem dob sepnutí sousedních napěťových vektorů se tedy realizuje výsledná poloha vektoru výstupního napětí $\underline{\mathbf{u}}^*$. Vložením nulových napěťových vektorů je pak měněna jeho velikost. Pro lepší pochopení je

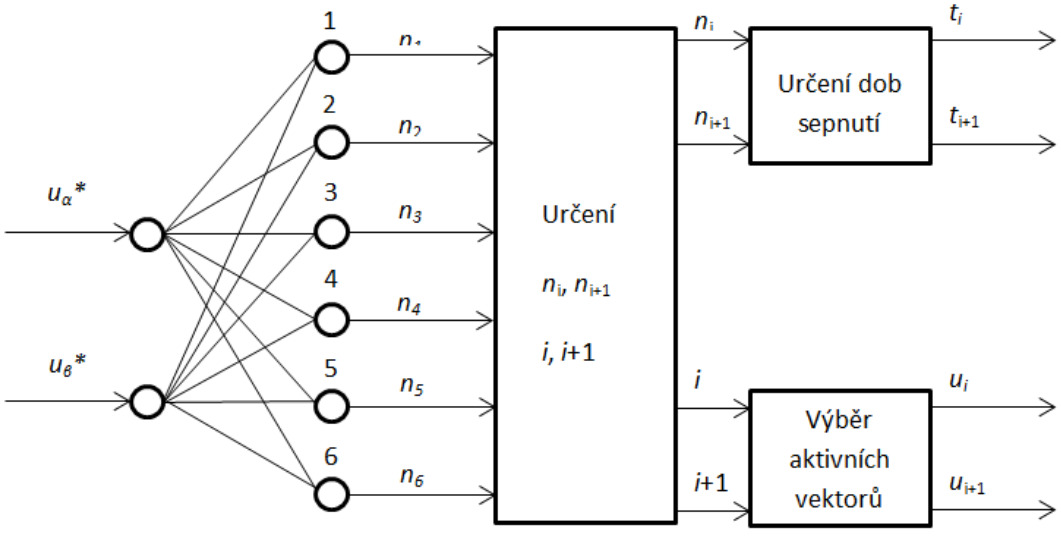
situace znázorněna na obrázku 3. Pro doby sepnutí aktivních napěťových vektorů platí vztah 5, kde T_1 je doba sepnutí prvního napěťového vektoru, T_2 je doba sepnutí druhého napěťového vektoru a T_0 je doba sepnutí nulového napěťového vektoru. Součet těchto tří dob pak tvoří spínací periodu.



Obr. 3 - Realizovatelné vektory výstupního napětí [1]

$$T_{sw} = T_1 + T_2 + T_0 \tag{5}$$

Hlavní výhodou využití kompetitivní neuronové sítě je to, že odpadá nutnost použití trigonometrických funkcí, které musí být pro určení aktivních napěťových vektorů a jejich časů sepnutí počítány v reálném čase. Tím je oproti tradiční vektorové PWM snížen obsah vyšších harmonických na výstupu měniče napětí. Na obrázku 4 je uvedeno blokové schéma této metody, které lépe vysvětluje její funkci.



Obr. 4 - Blokové schéma vektorové PWM využívající kompetitivní neuronovou síť [3]

Z obrázku 4 je zřejmé že do neuronové sítě vstupují složky [α, β] vektoru požadovaného výstupního napětí ve statorovém systému souřadnic. Tyto složky získáme transformací referenčních sinusových signálů, které jsou získány stejným způsobem jako v předchozích dvou metodách. Pro hloubku modulace platí vztah 3 a pro výsledné referenční napětí vztah 6.

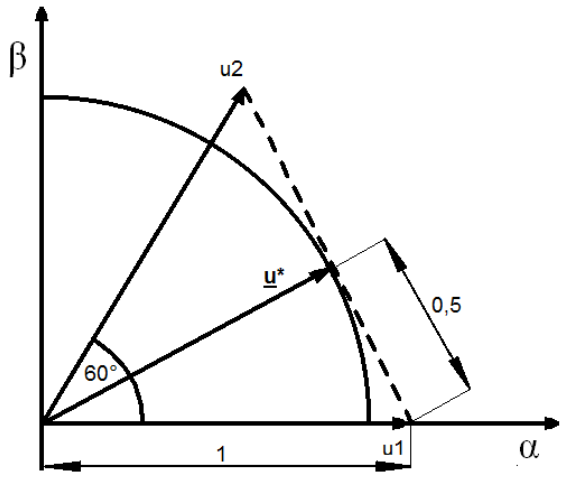
$$u(t)_{ref\ a,b,c} = m * \sin(\omega_{ref} t + 0, +\frac{2}{3}\pi, -\frac{2}{3}\pi) \tag{6}$$

$$u(t)_{\alpha=} 0{,}866 * u(t)_{ref\ a} \tag{7}$$

$$u(t)_{\beta} = \left(\frac{1}{\sqrt{3}} u(t)_{\alpha}\right) + \left(\frac{2}{\sqrt{3}} * 0{,}866 * u(t)_{ref\ b}\right) \tag{8}$$

Je nutno si povšimnout, že ve vztazích 7 a 8 přibyl koeficient 0,866. Ten je dán skutečností, že výsledný vektor **u*** se pohybuje po kružnici vepsané do šestiúhelníku na obrázku 3. Pokud by byla amplituda ponechána stejná jako v případě komparační PWM, docházelo by k přemodulování a tím k deformacím proudu. Podrobněji je situace naznačena na obrázku 5 a výpočet zmíněné konstanty je pak dán vztahem 9.

$$\left|\underline{\boldsymbol{u}}^*\right| = \sqrt{\alpha^2 - \beta^2} = \sqrt{1 - 0{,}5^2} = 0{,}866 \tag{9}$$



*Obr. 5 - výsledný vektoru **u*** pro danou velikost složek α, β*

Vypočtené sinusové signály jsou tedy transformovány do systému statorových souřadnic na složky [α, β] vektoru požadovaného výstupního napětí dle rovnic 7 a 8 a jsou zadány na vstupy neuronové sítě.

Dle vztahu 10, jehož odvození je uvedeno v pramenu [3] je pak realizována šestice neuronů, jejichž výstupy odpovídají reálné části skalárního součinu žádaného vektoru výstupního napětí **u***. Dvě největší hodnoty n_i a n_{i+1} jsou pak následně použity k určení správných spínacích vektorů a jejich dob sepnutí.

$$n_k = \cos\left[(k-1)\frac{\pi}{3}\right] u_\alpha{}^* - cos\left[(2k+1)\frac{\pi}{6}\right] u_\beta{}^* \tag{10}$$

Kde k = 1 – 6

Po dosazení do rovnice 10 získáme váhovou matici uvedenou v rovnici 11 pro kompetitivní neuronovou síť. Každým neuronem je pak realizován skalární součin váhového vektoru a žádaného vektoru výstupního napětí.

$$w = \begin{bmatrix} 1 & 0 \\ 1/2 & \sqrt{3}/2 \\ -1/2 & \sqrt{3}/2 \\ -1 & 0 \\ -1/2 & -\sqrt{3}/2 \\ 1/2 & -\sqrt{3}/2 \end{bmatrix} \tag{11}$$

Po výběru dvou nejvyšších výsledků neuronů n_i a n_{i+1} a po odvození uvedeném v [3] získáme vztahy pro výpočet výsledných dob sepnutí t_i a t_{i+1}.

$$t_i = (\frac{2T_{sw}}{3})(2n_i - n_{i+1}) \tag{12}$$

$$t_{i+1} = (\frac{2T_{sw}}{3})(2n_{i+1} - n_i) \tag{13}$$

Doba sepnutí nulového vektoru je pak dána vztahem 14.

$$T_0 = T_{sw} - t_i - t_{i+1} \tag{14}$$

Se získanou váhovou maticí a vztahy 12 a 13 lze vypočíst potřebné parametry spínaných vektorů výstupního napětí a pomocí DSP je realizovat.

3 Popis důležitých parametrů použitého hardwaru

Při psaní jakéhokoliv aplikačního softwaru je nezbytné se nejprve seznámit s hardwarovou strukturou řízeného zařízení. V případě již zhotoveného řídicího systému s funkční kostrou programu, je nutno znát vstupní logiku budičů výkonových prvků měniče, vlastnosti využívaných periferií DSP a vlastnosti DSP jako takového, což je předmětem této kapitoly.

3.1 Budiče výkonových prvků řízeného NMK

V použitém měniči jsou osazeny budiče CONCEPT 6SD106EI, které umožňují spínací frekvenci až 100 kHz [6] s odezvou 5-10 µs. Experimentálně bylo ověřeno, že výkonové spínače připojené k budičům jsou sepnuty při přivedení log. 0 na příslušný vstup budiče.

3.2 DSP TMS320F28335

Vybrané algoritmy řízení jsou realizovány na digitálním signálovém procesoru TMS320F28335, běžícím na taktovací frekvenci 150 MHz. V tabulce 1 jsou uvedeny vybrané parametry DSP.

Tabulka 1 - vybrané parametry DSP TMS320F28335 [5]

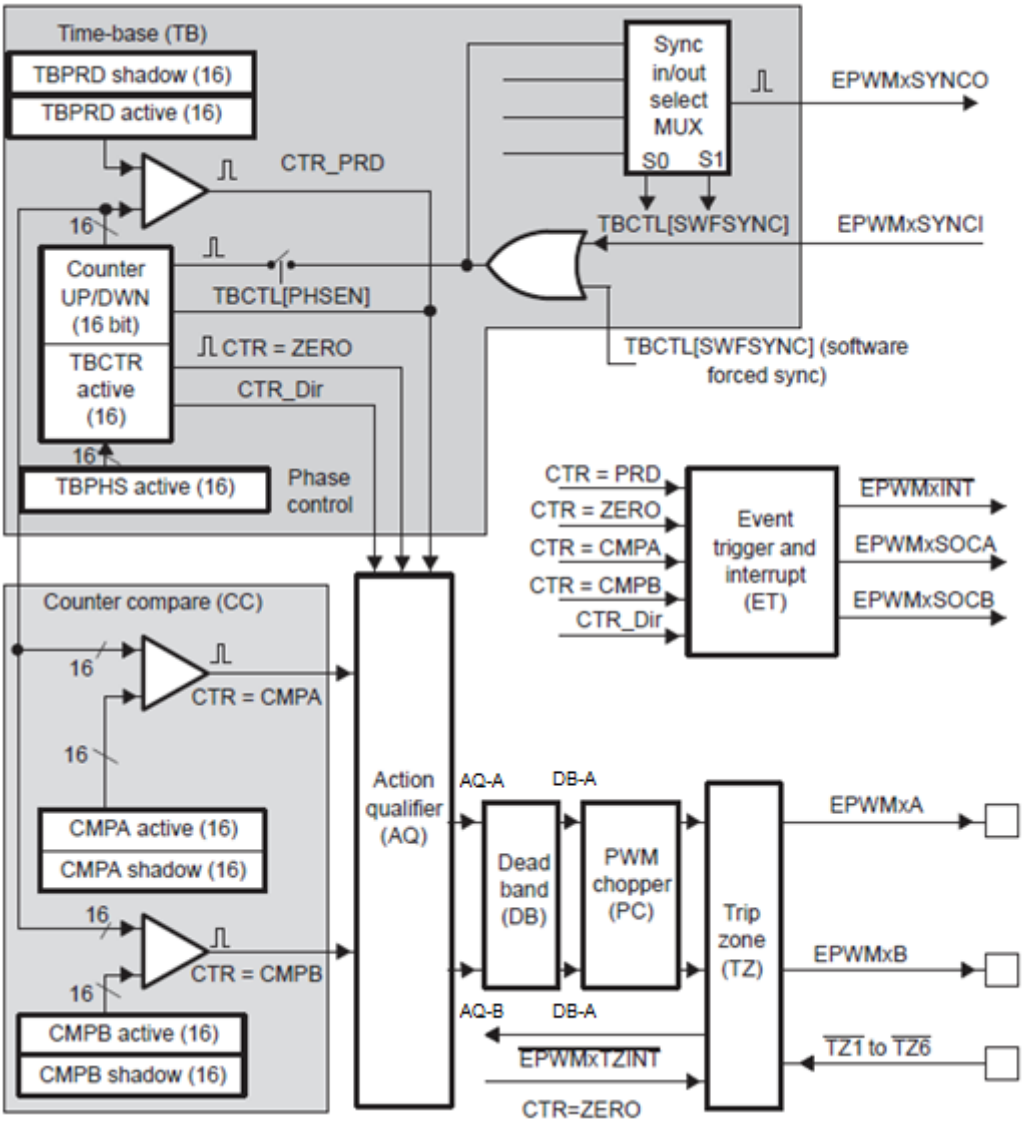
Napájecí napětí [V]	3,3
Taktovací frekvence[MHz]	150
Flash paměť	16x256k
SARAM paměť	16x34k
A/D převodník	12-bit, doba převodu 80 ns, 16 kanálů
ePWM	6x
CPU	32-bit IEEE-754 Single-Precision Floating-Point Unit

Jelikož řídicí systém spolu s DSP byly dodány i s kostrou programu, která realizuje základní nastavení periferií, budou dále uvedeny jen periferie DSP, které vyžadují vlastní nastavení pro generování PWM.

3.3 ePWM jednotka DSP TMS320F28335

Další vysvětlení chování jednotky ePWM se bude odkazovat na obrázek 6, na němž je naznačeno blokové schéma jednotky ePWM. DSP TMS320F28335 obsahuje celkem 6 identických jednotek ePWM z nichž pro naše účely postačí pouze první tři.

ePWM jednotka se skládá z jednotky časové základny TB, komparační jednotky CC, vyhodnocovací jednotky AQ, generátoru ochranné doby DB, chopperu PC, spouštěče událostí ET a jednotky TZ, která se stará o povolení výstupních pulzů. Jednotlivé bloky, na nichž závisí algoritmy, které budou následně implementovány, budou v následujících podkapitolách podrobněji rozebrány. Protože některé jednotky nebyly nijak nastavovány, buďto z důvodu, že jejich defaultní nastavení bylo vyhovující, nebo nebyla jejich funkce vyžadována, nebude se jimi tato práce dále zabývat.



Obr. 6 - Blokové schéma ePWM jednotky[4]

3.3.1 Časová základna TB

Jednotka časové základny TB se skládá z čítače counter UP/DOWN, který, jak již jeho název napovídá, podporuje čítání směrem nahoru i dolů. Čítání je při směru čítání nahoru zastaveno po dosažení hodnoty zapsané v registru TBPRD nebo při čítání dolů při dosažení nuly. Čítač je dle nastavení pak obnoven do výchozí hodnoty, nebo změní směr čítání a zároveň je generován impulz dosažení hodnoty v registru TBPRD, nebo nuly. Registr TBPRD je stínován, to znamená, že při zápisu do něj, je zapsanou hodnotou nejprve naplněn stínový registr a až po splnění nastaveních podmínek (např. vynulování čítače) je do aktivního registru zkopírována hodnota z registru stínového.

Čítač může být synchronizován impulzem přicházejícím z jiné jednotky ePWM, tato funkce je důležitá především při využití více jednotek ePWM najednou a zajišťuje tak jejich synchronizovaný chod. Jedna z jednotek je vždy nastavena bitem PHSEN jako master, ostatní pak od ní přijímají synchronizační pulzy.

Čítač obsahuje i registr fázového posunu PHSEN, ten ale pro naše účely není využit.

3.3.2 Komparační jednotka CC

Komparační jednotka se skládá ze dvou nezávislých komparátorů s vlastními registry CMPA a CMPB. I tyto registry jsou stejně jako TBPRD řešeny jako stínové registry a podmínky pro jejich přepis do registrů aktivních lze nastavit v řídícím registru komparační jednotky CMPCTL.

Komparátory generují impulz při rovnosti aktuálního stavu čítače obsaženého v registru TBCTR se zadanou hodnotou v příslušném registru CMPA nebo CMPB. Tyto impulzy pak společně s impulzy z jednotky časové základy vyhodnocuje vyhodnocovací jednotka.

3.3.3 Vyhodnocovací jednotka AQ

Do vyhodnocovací jednotky vstupují signály z časové základny a komparační jednotky. V závislosti na nastavení registrů AQCTLA a AQCTLB je definováno chování výstupních signálů AQ-A a AQ-B, které následně vstupují do generátoru bezpečnostní doby. Z blokového schématu je zřejmé, pro které podněty lze chování jednotky nastavit. Výstupy jednotky následně pokračují do generátoru bezpečnostních dob, jehož chování je popsáno v kapitole 3.3.5.

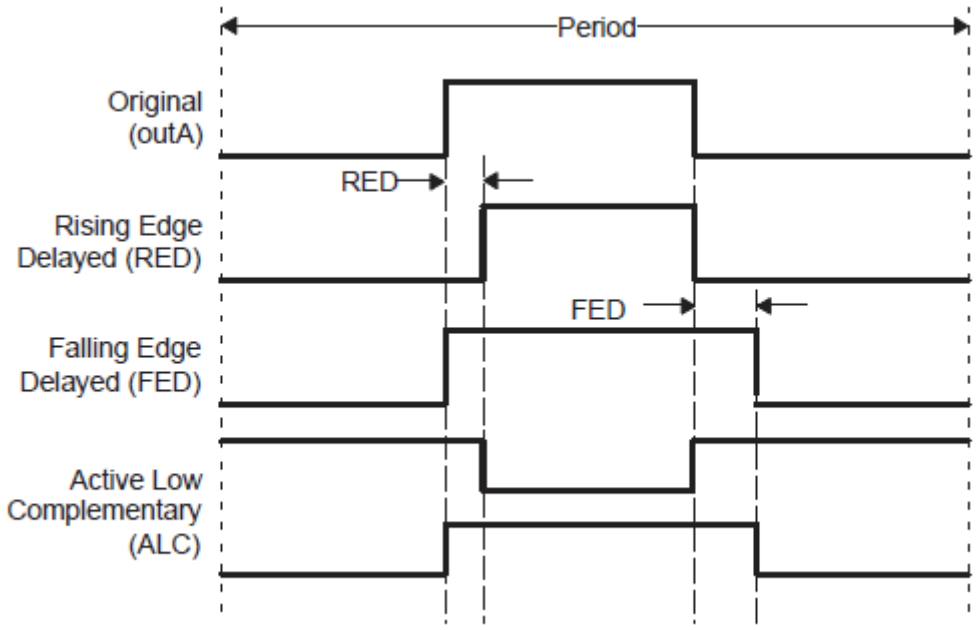
3.3.4 Spouštěč událostí ET

Jednotka spouštěče událostí generuje příznak přerušení a podněty pro další periferie, jako například spuštění A/D převodníku na základě nastavení registru ETPS. Přerušení od ePWM jednotek v tomto případě není žádoucí a proto je zakázáno. V registru ETPS lze nastavit, na který z podnětů bude jednotka reagovat a počet podnětů požadovaných k vygenerování pulzu na výstupech SOCA a SOCB. Po splnění zadaných podmínek je vygenerován pulz na příslušném výstupu, na který pak reagují další periferie podle jejich vlastního nastavení. V tomto případě, je touto periferií A/D převodník, který po zaznamenání pulzu započne převod a po jeho dokončení vyvolá přerušení. Dá se tedy říct, že nastavením chování výstupů SOCA a SOCB spouštěče událostí, lze v tomto případě přímo ovlivňovat podmínky spuštění převodu a tím pádem i přerušení.

3.3.5 Generátor bezpečností doby DB

Funkce tohoto modulu je zřejmá z jeho názvu. Mimo bezpečnostní doby náběžných a sestupných hran výstupního signálu, které jsou nastavovány jako počet hodinových pulzů časové základny v registrech DBRED (rising edge) a DBFED (falling edge), lze taky nastavit samotné chování výstupů jednotky v závislosti na vstupním signálu pomocí registrů DBCTL.

Pro účely řízení můstkového střídače se jeví, jako výhodné tuto jednotku použít v komplementárním módu, což znamená, že výstup DB-B je negací výstupu DB-A, která je navíc opožděna o dobu definovanou registry DBRED a DBFED. Pro tento způsob nastavení generátoru bezpečnostní doby je zapotřebí pouze signálu AQ-A. Chování tohoto zpoždění je definováno opět v registru DBCTL, kde je nutno nastavit v jaké logické úrovni budič aktivuje výkonový spínací prvek. V našem případě je budič aktivní v log. 0, čemuž odpovídá následující průběh spínání výstupů EPWMxA a EPWMxB uvedený na obrázku 7.



Obr. 7 - Průběh spínání výkonových spínačů v jedné větvi pro budič aktivní v log. 0[4]

4 Popis nastavení DSP TMS320F28335

V této kapitole bude podrobně rozebráno nastavení periferií DSP, na základě poznatků získaných kapitole 3. Nyní je tedy kladeno za cíl nastavit periferie DSP tak, aby plnily funkci požadovanou pro realizaci metod řízení výstupního napětí NMK.

4.1 Časování přerušení

Základním požadavkem pro spolehlivé generování pulzů řídících chod měniče, je pevně definovaná perioda přerušení, v jehož obsluze budou vypočteny požadované výstupní parametry a následně bude realizována spínací kombinace výkonových spínačů měniče.

Jak již bylo zmíněno v kapitole 3.3.4, přerušení je vyvoláváno od A/D převodníku po dokončení převodu. A/D převodník je spouštěn výstupním pulzem spouštěče událostí ET. Pro definování periody a podmínek přerušení je tedy nutno nejprve nastavit tento modul.

Dle manuálu k modulu ePWM [4] budou tedy do registrů jednotky ePWM1 zapsány následující hodnoty.

Zápisem hodnoty 1 do tohoto registru je povolen výstup SOCA jednotky ET

```
EPwm1Regs.ETSEL.bit.SOCAEN = 1;
```

Zápisem hodnoty 2 do tohoto registru je jako podmínka pro vyslání pulzu na výstup SOCA nastavena rovnost registrů TBCTR a TBPRD – jinými slovy tedy dosažení maximální nastavené hodnoty čítače časové základny.

```
EPwm1Regs.ETSEL.bit.SOCASEL = 2;
```

Dále je třeba zajistit, aby generování pulzu SOCA proběhlo při každém splnění nastavené podmínky.

```
EPwm1Regs.ETPS.bit.SOCACNT = 1;
EPwm1Regs.ETPS.bit.SOCAPRD = 1;
```

Protože nic jiného než A/D převodník již není třeba v závislosti na vnitřních stavech jednotky ePWM spouštět, je výstup SOCB vyřazen.

```
EPwm1Regs.ETSEL.bit.SOCBEN = 0;
```

Přerušení chceme vyvolávat pouze od jedné jednotky ePWM, proto je nastavena pouze jednotka ePWM1. Nyní jsou pevně definovány podmínky, za kterých dojde ke spuštění A/D převodníku a tedy i přerušení. Dalším logickým krokem je definování periody přerušení.

Pro výpočet periody časové základny TB je na základě způsobu jakým čítač čítá využit vztah 15 uváděný výrobcem v pramenu [4]. Čítač je dle následující hodnoty v registru TBCTL nastaven tak, že čítá nahoru a po dosažení hodnoty zadané v registru TBPRD je vynulován. Toto nastavení je identické i pro zbylé dvě jednotky.

```
EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;
```

$$T_{sw} = (TBPRD + 1) * T_{TBCLK} \tag{15}$$

Perioda časové základny T_{TBCLK} je odvozena od systémových hodin a nastavení před děliček časové základny a je dána vztahem 16. Dle [4] určíme z hodnot v řídících registrech časové základny nastavený dělící poměr. Tyto hodnoty jsou identické i pro zbylé dvě jednotky ePWM.

```
EPwm1Regs.TBCTL.bit.HSPCLKDIV = 2;
EPwm1Regs.TBCTL.bit.CLKDIV = 0;
```

$$T_{TBCLK} = \frac{HSPCLKDIV * CLKDIV}{f_{sys}} = \frac{4*1}{150*10^6} = 26{,}666\,ns \tag{16}$$

Bylo zvoleno, že realizované metody budou moci pracovat se třemi různými spínacími frekvencemi f_{sw} a to 1 kHz, 5 kHz a 10 kHz. Po dosazení výsledku ze vztahu 16 do vztahu 15 a vyjádření TBPRD získáme vztah 17 pro výslednou hodnotu TBPRD pro tyto spínací frekvence.

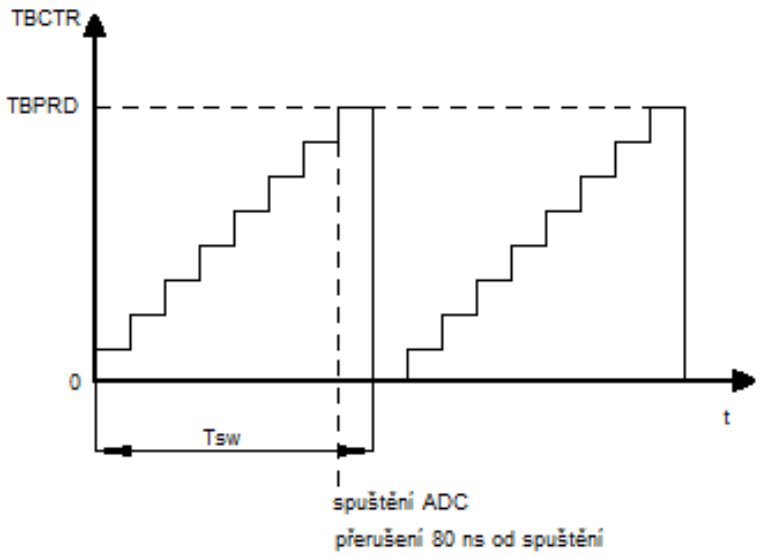
Vztah byl následně experimentálně doladěn dle osciloskopu, aby byla výstupní spínací frekvence co nejpřesnější. Hodnoty TBPRD odpovídající daným spínacím frekvencím jsou uvedeny v tabulce 2.

$$TBPRD = \frac{\frac{1}{f_{sw}}}{T_{TBCLK}} - 2 \tag{17}$$

Tabulka 2 - hodnoty TBPRD pro zvolené spínací frekvence

TBPRD [-]	f_{sw} [kHz]
37498	1
7498	5
3748	10

Pro lepší představu, jak takto nastavený systém přerušení pracuje, je jeho funkce znázorněna na obrázku 8.



Obr. 8 - Grafické znázornění časování přerušení

4.2 Nastavení jednotek ePWM

Po úspěšném nastavení systému přerušení je dalším krokem nastavení chování jednotek ePWM. Protože chování všech jednotek je identické, bude uvedeno nastavení pouze první jednotky jako příklad.

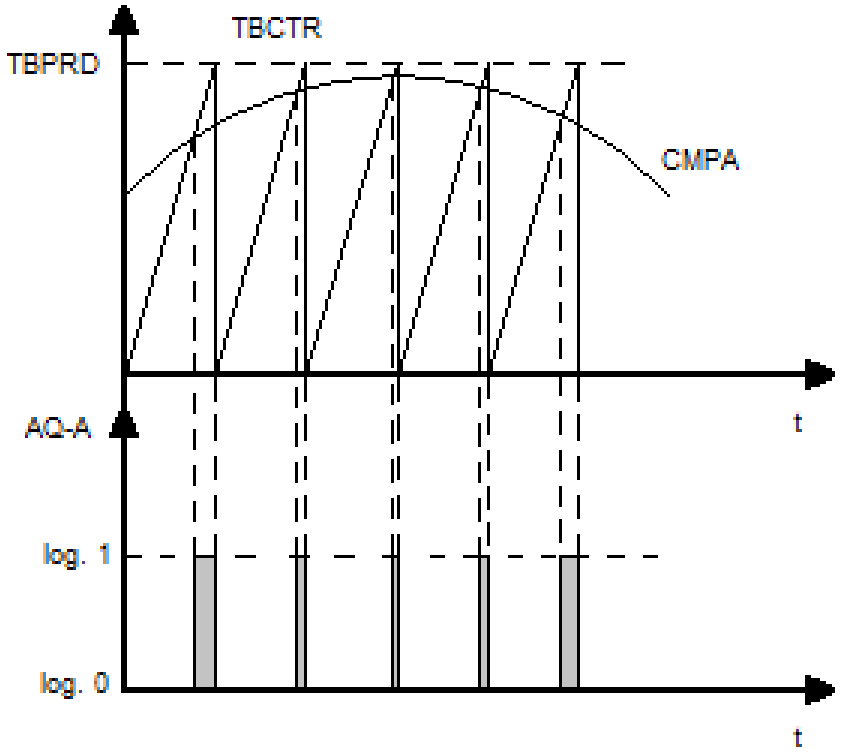
```
EPwm1Regs.TBPHS.half.TBPHS = 0; //vyrazeni registru fazoveho posunu
EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE; // Master modul
EPwm1Regs.TBCTL.bit.PRDLD = TB_SHADOW;
EPwm1Regs.TBCTL.bit.SYNCOSEL = TB_CTR_ZERO; // synch pulz posilan v 0
EPwm1Regs.TBCTL.bit.HSPCLKDIV = 2;
EPwm1Regs.TBCTL.bit.CLKDIV = 0;
EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;
EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;
EPwm1Regs.AQCTLA.bit.ZRO = AQ_CLEAR;
EPwm1Regs.AQCTLA.bit.CAU = AQ_SET;
EPwm1Regs.AQCTLA.bit.CAD = AQ_NO_ACTION;
EPwm1Regs.AQCTLA.bit.CBU = AQ_NO_ACTION;
EPwm1Regs.AQCTLA.bit.CBD = AQ_NO_ACTION;
EPwm1Regs.DBCTL.bit.OUT_MODE = DB_FULL_ENABLE; // povoleni DB
EPwm1Regs.DBCTL.bit.POLSEL = DB_ACTV_LOC; // Komplementarni aktivni v 0.
EPwm1Regs.DBFED = 125; // FED = 125 TBCLK
EPwm1Regs.DBRED = 125; // RED = 125 TBCLK
```

Jediným rozdílem v nastavení zbylých dvou jednotek je bit PHSEN, který je ve zbylých dvou jednotkách povolen. To znamená, že se chovají jako podřízené a synchronizují se pulzem vysílaným jednotkou ePWM1.

Jednotky jsou tedy nastaveny tak, že se pro zápis nových hodnot do komparačních registrů a registru TBPRD využívá stínových registrů. Tím je zajištěno, že hodnoty nebudou přepsány v době, kdy probíhá čítání čítače časové základny, ale až při příštím vynulování čítače.

Chování vyhodnocovací jednotky je nastaveno tak, že se používá pouze výstup AQ-A. AQ-B není třeba, protože správné spínání výstupů ePWMxA a ePWMxB v aktuálním nastavení zajišťuje generátor bezpečnostní doby pouze na základě signálu AQ-A. pro snazší pochopení nastavení vyhodnocovací jednotky je její chování znázorněno na obrázku 9.

Generátory bezpečnostní doby jsou nastaveny pro budiče aktivní v log. 0 a jejich chování popisuje obrázek 7. Doby zpoždění RED a FED jsou nastaveny na 3,3 µs.



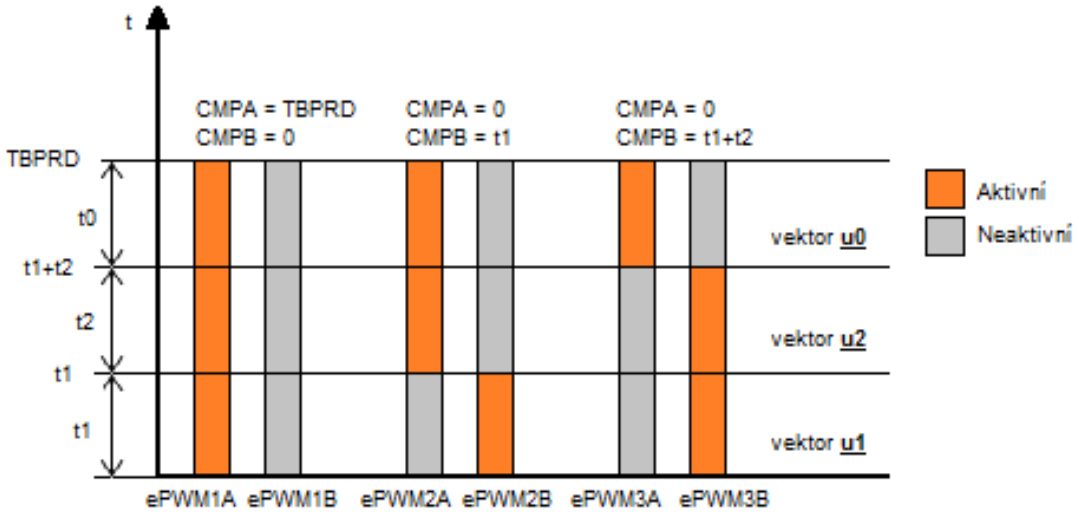
Obr. 9 - Grafické znázornění výstupu AQ-A v závislosti na nastavení jednotky ePWM

Z obrázku 9 je zřejmé, že registr CMPA bude pro každou periodu časové základny obsahovat vzorek referenčního signálu u_{ref}. Výpočet těchto vzorků a jejich zadávání do komparačního registru je náplní kapitoly 5.

Trochu jiný případ nastává při využití vektorové PWM, která vyžaduje odlišné nastavení jednotek ePWM. Nastavení jednotek se změní následujícím způsobem a je identické i pro zbylé dvě jednotky.

```
EPwm1Regs.AQCTLA.bit.CAU = AQ_SET;
EPwm1Regs.AQCTLA.bit.CBU = AQ_CLEAR;
```

Z uvedeného je na první pohled zřejmé, že nyní se bude využívat i druhého komparátoru komparační jednotky. Ač je tato změna nepatrná, má velký dopad na funkci jednotky jako takové a na způsob jakým se k ní bude přistupovat. Pro pochopení zamýšlené funkce a důvodu této změny v nastavení jednotek je na obrázku 10 znázorněn vzorový způsob realizace vektoru výstupního napětí **u***, který se nachází v sektoru I.



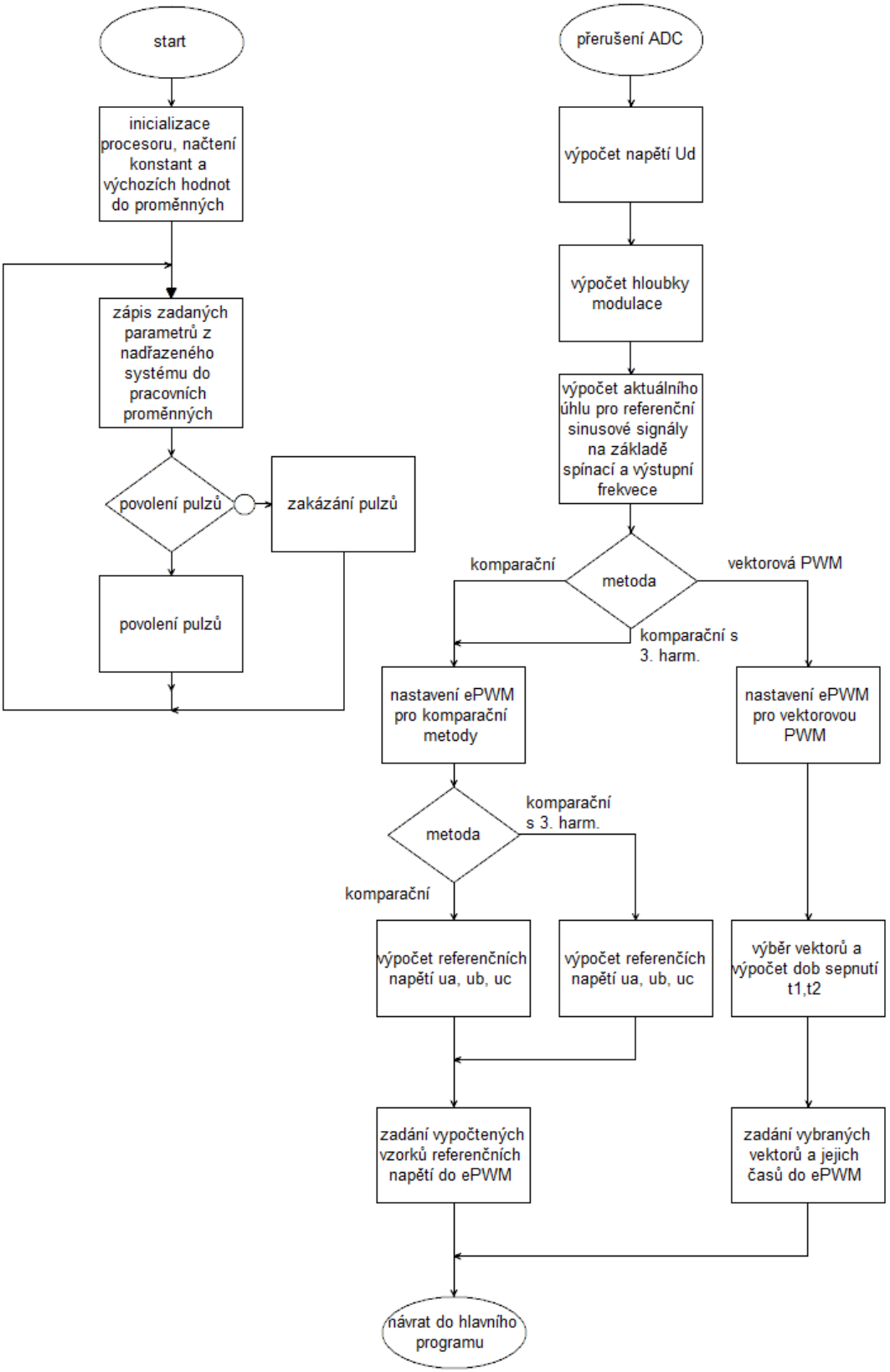
Obr. 10 - grafické znázornění realizace vektoru výstupního napětí s budiči aktivními v log. 0

5 Popis aplikačního softwaru pro DSP

V této kapitole je podrobně rozebírána funkce aplikačního softwaru realizujícího vybrané metody řízení výstupního napětí. Protože pro komunikaci s nadřazeným systémem je využito metody přímého adresování paměti, není v procesoru nijak zvlášť řešena komunikace. Ačkoliv to komunikaci s DSP značně zjednodušuje, je třeba mít na paměti, že k zápisu do proměnných může dojít ve kterémkoliv bodě běhu programu. Proto je nutno parametry, které jsou z nadřazeného systému přepisovat z pomocných proměnných do proměnných, které jsou v programu dále využívány v určité části programu, aby nedošlo nevhodným přepisem k jeho nestabilitě. Ideálním místem k takovéto operaci je hlavní smyčka programu.

V následujících podkapitolách jsou rozebrány způsoby realizace jednotlivých vybraných metod řízení výstupného napětí NMK. Kompletní zdrojový kód pro DSP a sestavené uživatelské prostředí jsou k dispozici v příloze I na dodaném CD.

5.1 Vývojový diagram a popis činnosti aplikačního software



Obr. 11 - Vývojový diagram aplikačního software

Na obrázku 11 je znázorněn vývojový diagram aplikačního software procesoru. Po zapnutí napájení jsou nastaveny a spuštěny všechny periferie a jsou zadány výchozí hodnoty do všech používaných globálních proměnných. V tu chvíli už začínají pracovat i jednotky ePWM. Dokud nedojde k povelu z nadřazeného systému, nejsou povoleny výstupní pulzy. Část kódu blokující výstupní pulzy byla dodána jakou součást kostry aplikačního softwaru pro DSP a nebylo s ní nijak manipulováno.

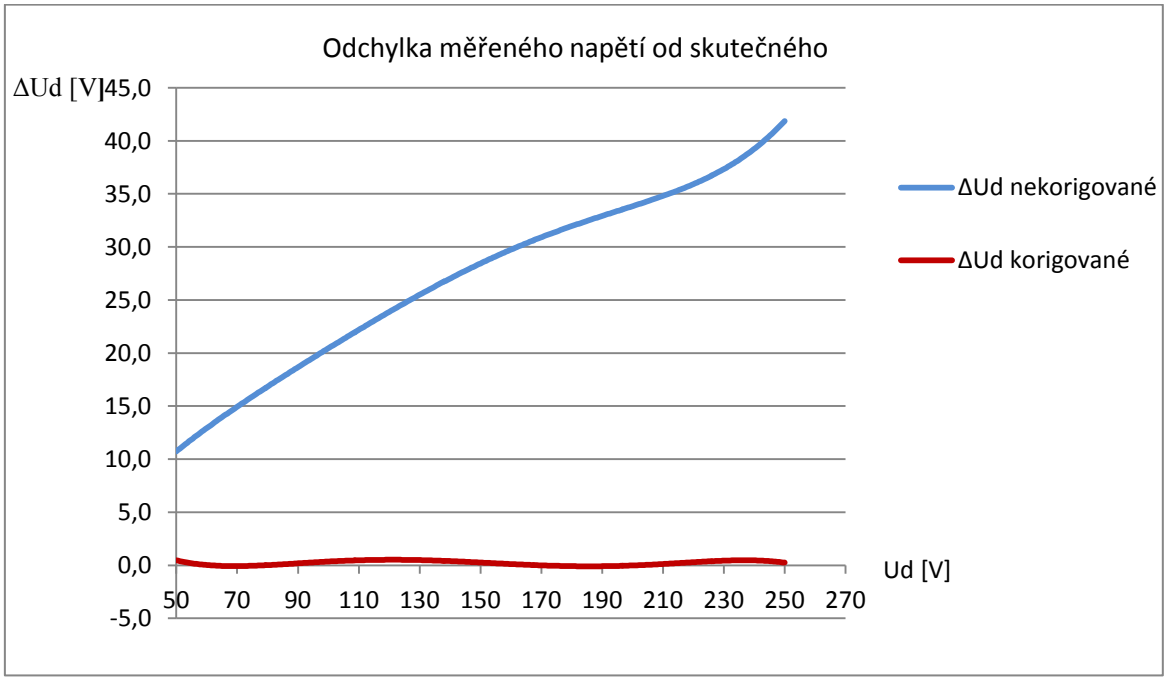
Při přetečení čítače časové základny jednotky ePWM1 je spuštěn převod A/D převodníku, který změří napětí meziobvodu a po dokončení převodu je vyvoláno přerušení.

V přerušení je z hodnoty A/D převodníku vypočtena průměrná hodnota skutečného napětí za posledních 16 vzorků. Vzorec pro získání napětí UMeziOActual byl dodán spolu s kostrou programu. Realizace výpočtu, je zřejmá z následujícího kódu.

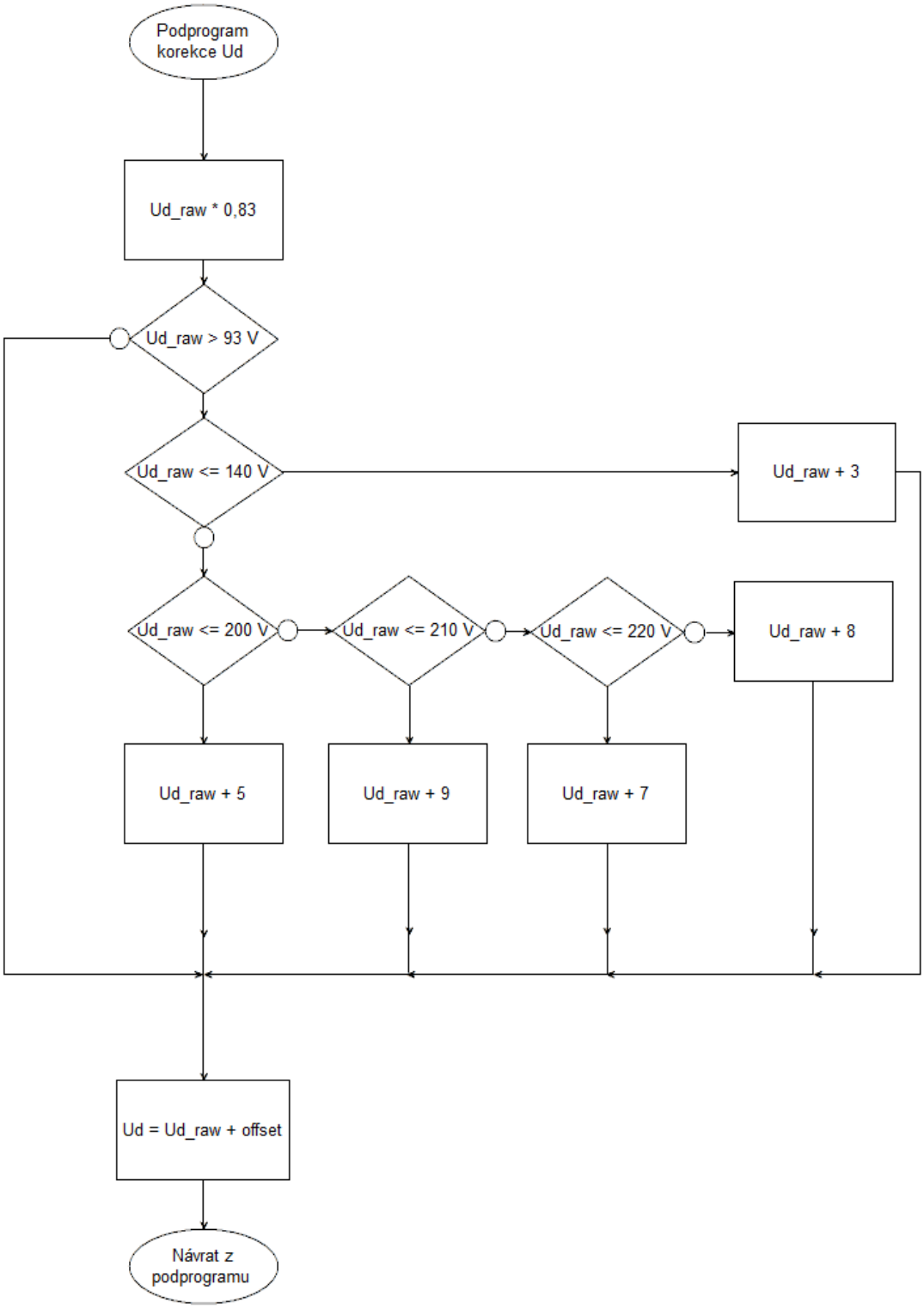
```
UMeziOActual =  (-1) * (((float32) AdcMirror.ADCRESULT0) - 2071) / 3.18;
UMeziOSuma = UMeziOSuma + UMeziOActual;
Inkrement1++;

if (Inkrement1 > 16)
      {
      U_meziobvod_raw = 0.83*(UMeziOSuma / 16);
      UMeziOSuma = 0;
      Inkrement1 = 0;
      }
```

Takto získané napětí však bylo značně nepřesné především v oblasti 93 V – 250 V. Chyba měření byla potlačena vynásobením vypočteného napětí experimentálně získanou konstantou 0,83, stále však přetrvávala nelineární odchylka napětí, která byla vyřešena “ocejchováním” měřeného napětí. Původní přesnost měření a přesnost po zmíněných korekcích je srovnaná v grafu na obrázku 12. Na obrázku 13 je uveden vývojový diagram řešící korekci měření napětí.



Obr. 12 - Vliv korekce měřeného napětí na přesnost měření



Obr. 13 - Vývojový diagram korekce měřeného napětí

Na základě změřeného napětí meziobvodu a požadovaného výstupního napětí je vypočtena požadovaná hloubka modulace pro zadanou metodu řízení výstupního napětí následujícím způsobem. Rovněž je zde řešen chybový stav, který by mohl nastat při zadání amplitudy výstupního napětí vyšší, než je možné. Aby nedocházelo k přemodulování, je v takovém případě hloubka modulace pevně nastavena na hodnotu 1.

```
switch (metoda_akt)
	{
	case 0:                                  //komparacni
U_max_out = 0.5*U_meziobvod;
	break;

	case 1:                                  //komparacni s 3. harmonickou
U_max_out = 0.577*U_meziobvod;
	break;

	case 2:                                  //vektorova PWM
U_max_out = 0.577*U_meziobvod;
	break;
	}

if(U_zadane_rx >= U_max_out)
{
	hloubka_m = 1;
}
else
{
	hloubka_m = U_zadane_rx / U_max_out;
}
```

Následně je vypočten aktuální úhel, který je využíván pro následující výpočty referenčních napětí dle vztahu 18.

$$\omega = \omega_{t-1} + \frac{2\pi}{\frac{f_{sw}}{f_{out}}} \tag{18}$$

Aby nedocházelo k přetečení proměnné obsahující úhel ω je vždy při dosažení hodnoty 2π tato hodnota od aktuální hodnoty proměnné ω odečtena, jak je vidět v následujícím kódu. Rovněž je zamezeno zadání nulové, nebo menší frekvence.

```
if(fout_zadana<=0)
{
	fout_zadana=1;
}
deltat = fvz_zadana/fout_zadana;
deltaoemga = 2/deltat;

omega = omega + deltaomega;
if(omega>=2)
{
	omega=omega-2;
}
```

V kódu se nepočítá s hodnotami π ale pouze s jejich násobky. Násobek je násoben číslem π až při výpočtu vlastních referenčních napětí.

Po výpočtu úhlu ω následují realizace jednotlivých metod, které jsou předmětem následujících podkapitol.

5.2 Realizace komparační PWM

Následující kód realizuje vypočtení tří referenčních napětí a jejich následné zadání do jednotek ePWM. Pokud byla v přechozím průchodu obsluhou přerušení využívána metoda vektorové PWM, je splněna podmínka a příslušné registry ePWM jednotek jsou nastaveny pro použití komparačních PWM, což je popsáno v kapitole 4.2.

Protože do komparačních registrů lze zapisovat pouze kladná čísla, je ještě navíc nutno referenční signály posunout tak, aby při maximální hloubce modulace nabývaly hodnot 0 až 1.

Při zápisu referenčního signálu do komparačního registru je třeba brát zřetel na rozdílné datové typy. Vypočtená hodnota referenčního napětí je násobena obsahem proměnné pwmprd, která je identická s obsahem registru TBPRD. To znamená, že například při hodnotě sinusa = 0,5 a nastavené f_{sw} = 1 kHz, dojde ke komparaci ve chvíli, kdy čítač dosáhne hodnoty 18749.

Protože převodem z datového typu float32 na datový typ Uint16 dochází ke ztrátě číslic za desetinnou čárkou, může se stát, že z například z čísla 1,9 ve formátu float32 vznikne po převodu číslo 1 ve formátu Uint16. Právě z tohoto důvodu je k výsledku násobení přičteno číslo 0,5, čímž je zajištěno, že při převodu datových typů dojde zároveň ke správnému zaokrouhlení namísto “useknutí” informace za desetinnou čárkou.

```
if(metoda_flag)
{
InitEPwm(pwmprd, PWM_HSD, PWM_DIV, PWM_FED, PWM_RED);
}
sinusa = ((1+hloubka_m * sin (omega *PI))/2);
sinusb = ((1+hloubka_m * sin ((omega *PI)+PI23))/2);
sinusc = ((1+hloubka_m * sin ((omega *PI)-PI23))/2);
metoda_flag = false;

EPwm1Regs.CMPA.half.CMPA = (Uint16) ((pwmprd * sinusa)+0.5);
EPwm2Regs.CMPA.half.CMPA = (Uint16) ((pwmprd * sinusb)+0.5);
EPwm3Regs.CMPA.half.CMPA = (Uint16) ((pwmprd * sinusc)+0.5);
```

5.3 Realizace komparační PWM s přidáním třetí harmonické

Kód je v tomto případě obdobný jako v přecházejícím případě. Jediným rozdílem je přičtení 3. harmonické a z toho vyplývající násobení 1. harmonické koeficientem uvedeným v kapitole 2.2. Po získání součtu první a třetí harmonické je výsledný signál násoben hloubkou modulace a posunut tak aby nabýval hodnot 0 až 1. Zápis do komparačních registrů je totožný s předchozí metodou a proto ani není v následujícím kódu uveden.

```
if(metoda_flag)
      {
      InitEPwm(pwmprd, PWM_HSD, PWM_DIV, PWM_FED, PWM_RED);
      }
      temp_sinus = 0.166666667 * (sin (3* omega *PI));
      sinus1 = (temp_sinus + (1.1547*(sin (omega *PI))));
      sinus2 = (temp_sinus + (1.1547*(sin ((omega *PI)-PI23))));
      sinus3 = (temp_sinus + (1.1547*(sin ((omega *PI)+PI23))));
      sinusa = ((1+hloubka_m * sinus1)/2);
      sinusb = ((1+hloubka_m * sinus2)/2);
      sinusc = ((1+hloubka_m * sinus3)/2);
      metoda_flag = false;
```

5.4 Realizace vektorová PWM s využitím kompetitivní neuronové sítě

Následující kód v první řadě ověří, zda nebyla v přechozím průchodu obsluhou přerušení využívána jedna z komparačních metod a v případě, že ano, dojde k nastavení registrů ePWM, které je nutné pro realizaci metody vektorové PWM.

V dalším kroku jsou vypočtena referenční napětí, která jsou dále převedena na složky [α, β] statorového systému souřadnic. Složky [α, β] jsou při převodu násobeny koeficientem 0,866, jehož původ je uveden v kapitole 2.3. Dále jsou tyto složky využity neurony n[0] až n[5].

```
if(!metoda_flag)
{
	EPwm1Regs.AQCTLA.bit.CAU = AQ_SET;
	EPwm2Regs.AQCTLA.bit.CAU = AQ_SET;
	EPwm3Regs.AQCTLA.bit.CAU = AQ_SET;
	EPwm1Regs.AQCTLA.bit.CBU = AQ_CLEAR;
	EPwm2Regs.AQCTLA.bit.CBU = AQ_CLEAR;
	EPwm3Regs.AQCTLA.bit.CBU = AQ_CLEAR;
}
sinus1 = hloubka_m * (sin (omega*PI));
sinus2 = hloubka_m * (sin ((omega *PI)+PI23));
ualfa = 0.866*sinus1;
ubeta = 0.866*((0.577350269 * ualfa) + (1.154700538 * sinus2));
n[0] = ualfa;
n[1] = (0.5 * ualfa) - (0.866025403 * ubeta);
n[2] = ((-0.5) * ualfa) - (0.866025403 * ubeta);
n[3] = (-ualfa);
n[4] = ((-0.5) * ualfa) - ((-0.866025403) * ubeta);
n[5] = (0.5 * ualfa) - ((-0.866025403) * ubeta);
```

Pro výběr dvou nejvyšších vektorů je využito následujícího algoritmu. Smyčka **for** proběhne 6x a v jejím prvním průběhu je porovnána hodnota neuronu n[0] s hodnotou 0. Protože je tím splněna podmínka, že hodnota neuronu je vyšší než hodnota obsažená v proměnné max, dojde k zápisu hodnoty neuronu n[0] do proměnné max a do proměnné vektor1 je uložena hodnota 1. V dalším průběhu dochází ke stejnému porovnávání ale tentokrát již s hodnotou neuronu n[0]. Takto se porovnají všechny neurony a při dokončení smyčky zůstane v proměnné max uložena nejvyšší nalezená hodnota a v proměnné vektor1 číslo napěťového vektoru, kterému tato hodnota odpovídá.

Smyčka hledající druhý největší výsledek neuronů pracuje stejným způsobem, ale je v ní navíc přidána podmínka, která vylučuje první nejvyšší neuron, který byl nalezen v předchozí smyčce. Výstupem druhé smyčky je tedy hodnota druhého nejvyššího neuronu a jí odpovídající číslo napěťového vektoru.

Z takto získaných dvou nejvyšších neuronů jsou pak vypočteny časy sepnutí vektoru1 a vektoru2.

```
            max = 0;
            max2 = 0;
            for (it=0;it<6;it++)
            {
                  if (n[it]>max)
                  {
                        max = n[it];
                        vektor1 = it + 1;
                  }
            }
            for (jt=0;jt<6;jt++)
            {
                  if (n[jt]!=max)
                  {
                        if (n[jt]>max2)
                        {
                        max2 = n[jt];
                        vektor2 = jt + 1;
                        }
                  }
            }
            cas1 = (0.666666666 * Tvz) * ((2 * max) - max2);
            cas2 = (0.666666666 * Tvz) * ((2 * max2) - max);

      metoda_flag = true;
```

V následujícím kódu jsou realizovány samotné napěťové vektory. Podle čísla výsledného prvního vektoru je vybrána odpovídající spínací kombinace a příslušné vypočtené časy sepnutí jsou zapsány do pomocných proměnných, které svými názvy odpovídají komparačním registrům, do nichž následně bude proveden zápis. Podle čísla druhého vektoru jsou pak zadány časy pro jeho odpovídající spínací kombinaci opět do pomocných proměnných. Princip funkce zadávání vektorů napětí a jejich příslušných časů sepnutí již byl objasněn v kapitole 4.2. Protože část kódu realizující tuto operaci je poměrně rozsáhlá, je zde uvedena jako příklad jen část kódu realizující napěťový vektor u_1 jako první napěťový vektor a u_2 jako druhý napěťový vektor.

```
            switch (vektor1)
            {
                  case 1:
                        compa1 = cas1;
                        compa2 = 0;
                        compa3 = 0;
                  break;



            switch (vektor2)
            {
                  case 1:
                        compb1 = cas1;
                        compb2 = cas1;
                        compb3 = cas1+cas2;
                  break;
            }
```

Za zmínku stojí následující část kódu, která chrání komparační registry před přetečením a realizuje samotný zápis vypočtených časů z pomocných proměnných do těchto registrů.

Může nastat případ, kdy si budou hodnoty pomocných proměnných comba a compb rovny. V takovém případě by docházelo k nedefinovaným stavům, protože by oba komparátory signalizovaly rovnost a ve vyhodnocovací jednotce by tak nastaly požadavky na zapnutí i vypnutí výstupu AQ-A zároveň. Z tohoto důvodu jsou nejprve pomocné proměnné na tuto rovnost otestovány. Pokud se skutečně rovnají, pak to znamená, že daný výstup bude aktivní po celou dobu T_{SW} a do obou registrů je zapsána hodnota pwmprd+1,takže ke komparaci nikdy nedojde.

Pokud podmínka rovnosti není splněna, pak jsou vypočtené časy násobeny pomocnou proměnnou pwmprd, pro kterou platí totéž co v případě komparačních metod. Navíc zde přibývá násobení zadanou vzorkovací frekvencí, čímže jsou časy sepnutí prezentovány jako hodnoty nabývající 0 až 1, kterými je proměnná pwmprd násobena.. Následuje zaokrouhlení a převod na datový typ vhodný pro zápis do komparačních registrů stejně jako v případě metod komparačních PWM. Pro jednoduchost je opět uvedena jen část kódu realizující zápis do registrů jednotky ePWM1, zápis do zbylých dvou jednotek je obdobný.

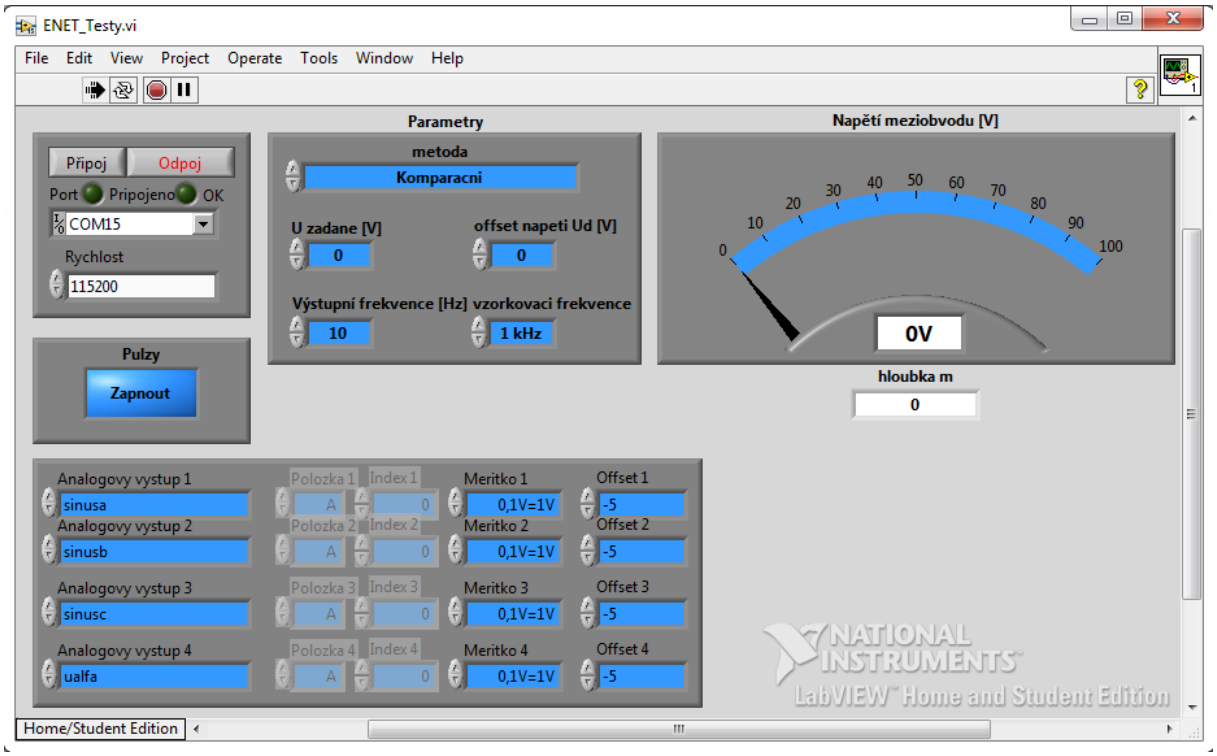
```
if(compa1==compb1)
{
EPwm1Regs.CMPA.half.CMPA = pwmprd+1;
EPwm1Regs.CMPB = pwmprd+1;
}
else
{
EPwm1Regs.CMPA.half.CMPA = (Uint16) ((fvz_zadana * pwmprd * compa1)+0.5);
EPwm1Regs.CMPB = (Uint16) ((fvz_zadana * pwmprd * compb1)+0.5);
}
```

6 Uživatelské rozhraní

Rozhraní pro komunikaci s uživatelem je realizováno v prostředí Labview a jeho čelní panel je znázorněn na obrázku 14. Protože uživatelské rozhraní bylo dodáno jakou součást kostry aplikačního softwaru pro DSP, bylo nutné je jen upravit, aby plnilo požadovaný účel a mělo požadovaný vzhled. Z tohoto důvodu bude popsán jen způsob komunikace s řídicím systémem, jehož pochopení bylo nezbytné k implementaci vlastních funkcí.

Panel umožňuje připojení k řídicímu systému a zobrazuje aktuální napětí meziobvodu spolu s vypočtenou hloubkou modulace. Panel dále umožňuje nastavení zapínání a vypínání výstupních pulzů z DSP, volbu použité metody řízení výstupního napětí, zadání amplitudy výstupního napětí, doladění offsetu měření napětí meziobvodu, nastavení výstupní frekvence napětí a volbu vzorkovací frekvence.

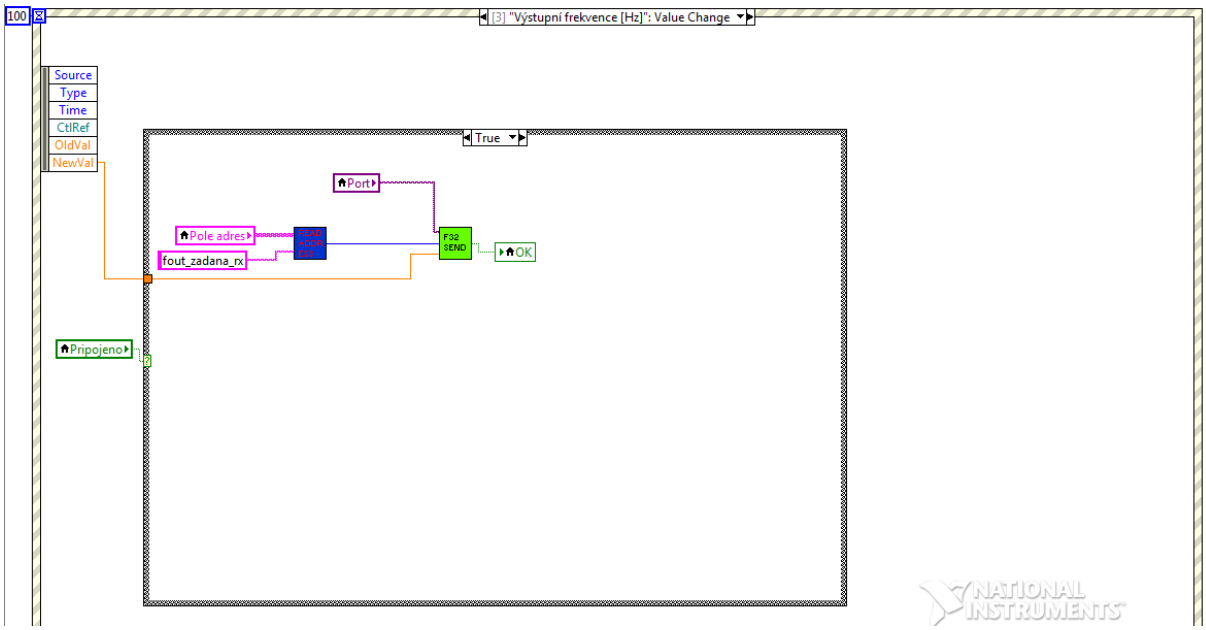
Ve spodní části panelu je pak blok umožňující výběr proměnných, které budou posílány na D/A převodník DSP, jejich měřítka a offsetu.



Obr. 14 - Čelní panel uživatelského prostředí

6.1 Komunikace s řídicím systémem

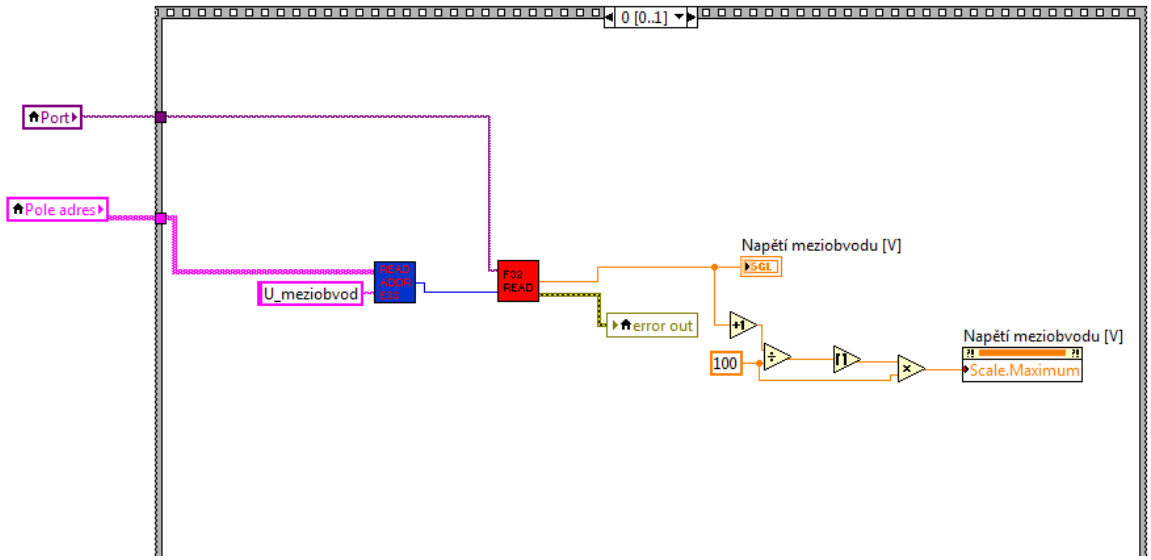
Jak již bylo zmíněno v kapitole 5, komunikace s DSP je řešena jako přímý zápis do paměťového prostoru DSP. Příklad zápisu například zadané výstupní frekvence je na obrázku 15.



Obr. 15 - Způsob odesílání dat do DSP

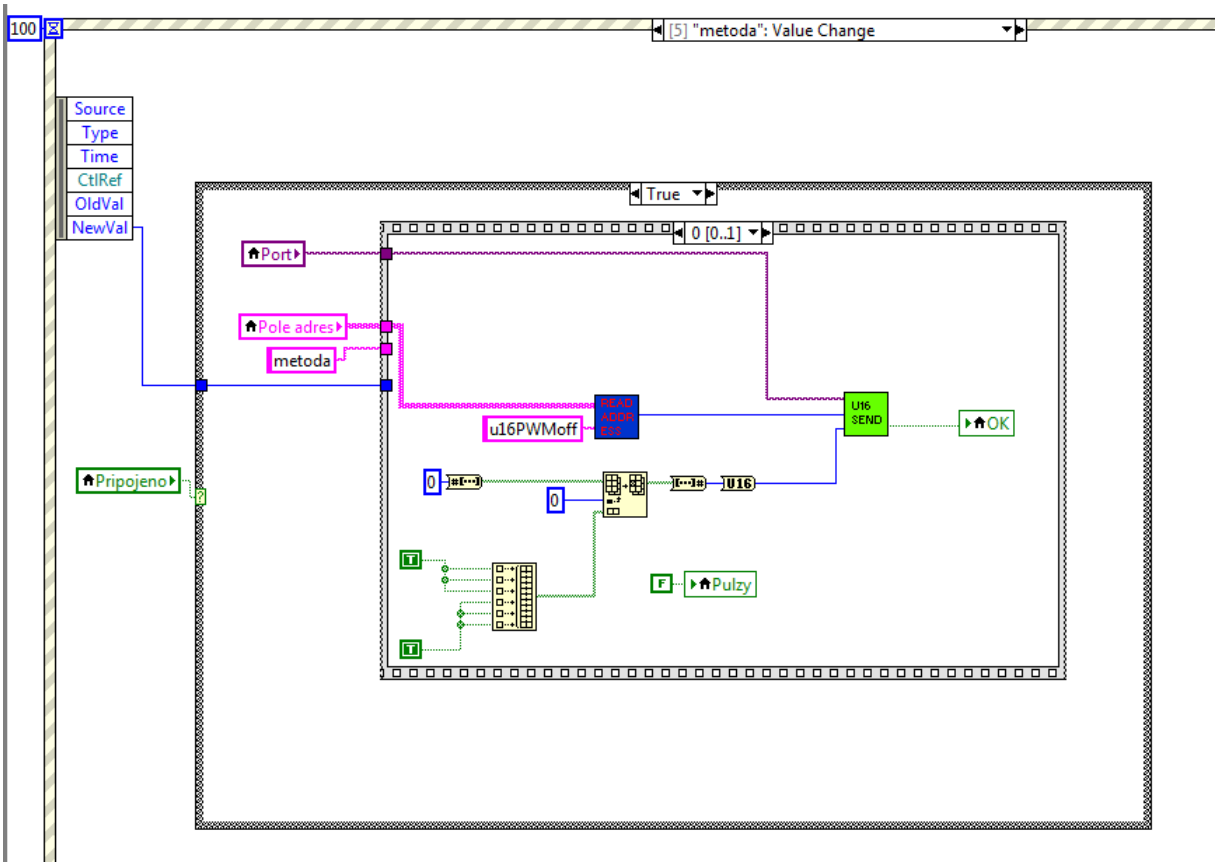
Při změně hodnoty políčka pro nastavení výstupní frekvence program skočí do rutiny odeslání nových dat řídicímu systému. Nejprve je v poli adres, které je generováno při kompilaci zdrojového kódu pro DSP nalezena adresa příslušející názvu proměnné fout_zadana_rx. Na tuto adresu je pak pomocí subVI F32 SEND zapsána hodnota, která byla do políčka na čelním panelu zadána.

V případě čtení dat z procesoru je situace obdobná. V programu je každých 100 ms generován timeout časovače, který je jednou z událostí, podobně jako změna hodnoty v některém z políček nastavujícím parametry. Při tomto timeoutu je zpracována rutina čtení dat, která je realizována jako filmová struktura – to znamená, že se vyčítají data z jednotlivých adres zvlášť. Pro lepší znázornění situace je na obrázku 16 uvedeno čtení měřeného napětí meziobvodu. Z obrázku je zřejmé, že přístup k paměťovému místu v DSP je realizován stejně jako při zápisu. Pro čtení je použito subVI F32READ.

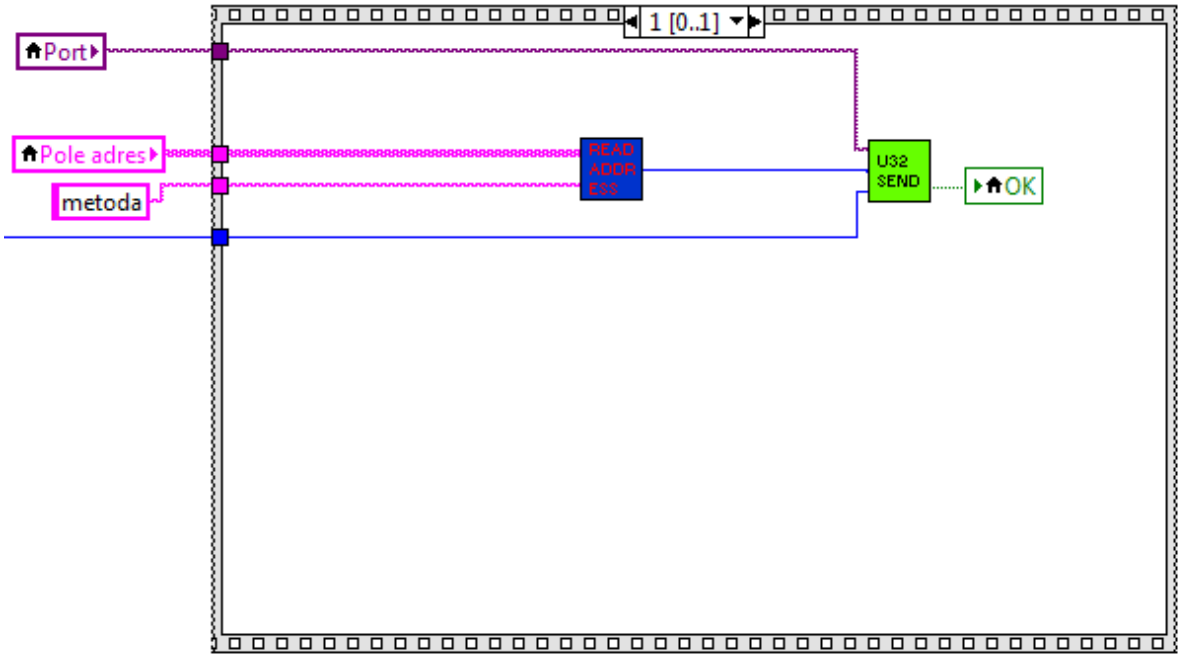


Obr. 16 - Způsob čtení dat z DSP

Na obrázcích 17 a 18 je znázorněno, jak probíhá bezpečné přepnutí metody řízení výstupního napětí měniče. Při změně metody jsou nejdříve zakázány výstupní pulzy z DSP a následně je do paměti DSP zapsána hodnota označující zvolenou metodu.



Obr. 17 - vypnutí výstupních pulzů při změně metody



Obr. 18 - Zadání nové metody řízení výstupního napětí

7 Experimentální výsledky na laboratorním stanovišti

V této kapitole je demonstrována správná funkce realizovaného aplikačního softwaru a ověření, zda experimentální výsledky korespondují s teoretickými předpoklady uvedenými v kapitole 2. Měření průběhů bylo provedeno čtyř kanálovým osciloskopem LeCroy Wave Surfer 424. V tabulce 3 jsou uvedeny převodní konstanty měřících sond a podmínky, za kterých bylo měření provedeno. Měření bylo provedeno na asynchronním motoru s nevyvedeným uzlem, jehož vybrané parametry jsou uvedeny v tabulce 4.

Tabulka 3 - Převodní konstanty a podmínky měření

-	Komparační metody	Metoda vektorové PWM
Napětí meziobvodu U_d	100 V	100 V
Převodní poměr napěťové sondy	100/1	100/1
Převodní poměr proudové sondy	100/1	100/1
Kanál1	$u(t)_{ref\,a}$	$u(t)_{\alpha}$
Kanál 2	$u(t)_{ref\,b}$	$u(t)_{\beta}$
Kanál 3	Napětí fáze a	Napětí fáze a
Kanál 4	Proud fáze b	Proud fáze b

Tabulka 4 - Vybrané parametry motoru

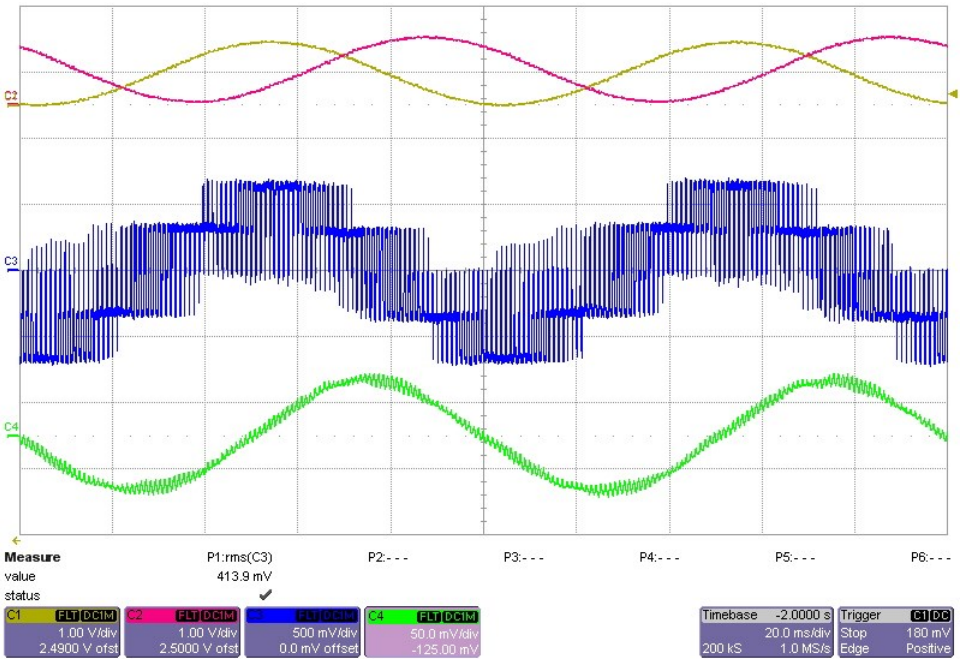
Jmenovitý výkon	2,7 kW
Jmenovitý cosϕ	0,75
Jmenovité otáčky	1360 *ot/min*
Jmenovité statorové napětí	380/220 *V*, zapojení do hvězdy
Jmenovitý statorový proud	7,51 *A*
Odpor fáze statoru	1,83 Ω - 2,10 Ω

7.1 Komparační PWM

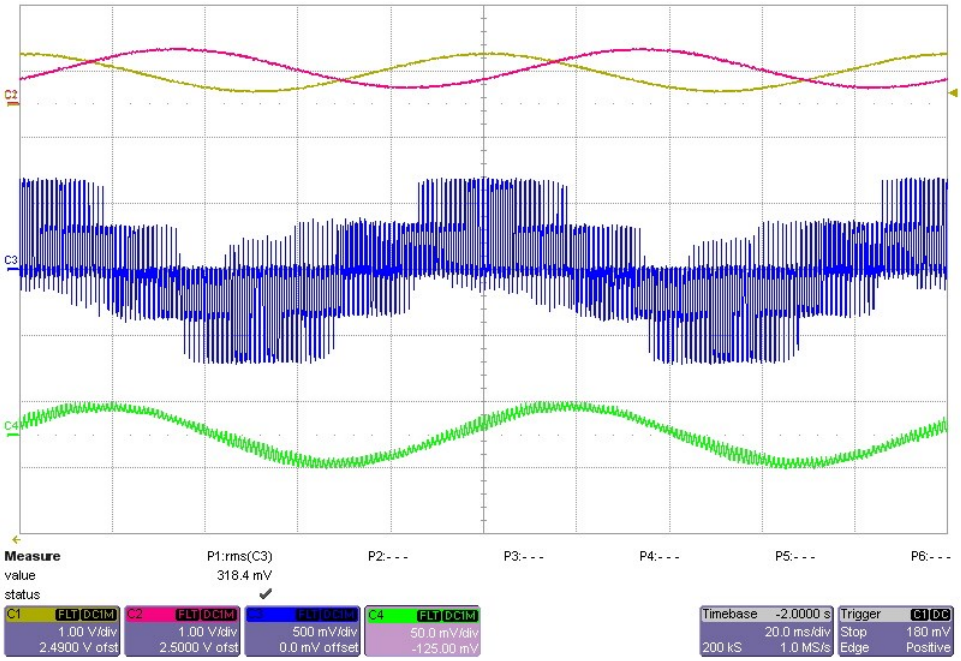
Průběhy pro různé parametry výstupního signálu jsou uvedeny na obrázcích 19 až 22. V tabulce 5 jsou pak shrnuty získané výsledky. Na obrázku 23 je zaznamenán průběh pro f_{sw} = 10 kHz a na obrázku 24 pak detail modulace výstupního napětí.

Tabulka 5 - Shrnutí naměřených výsledků

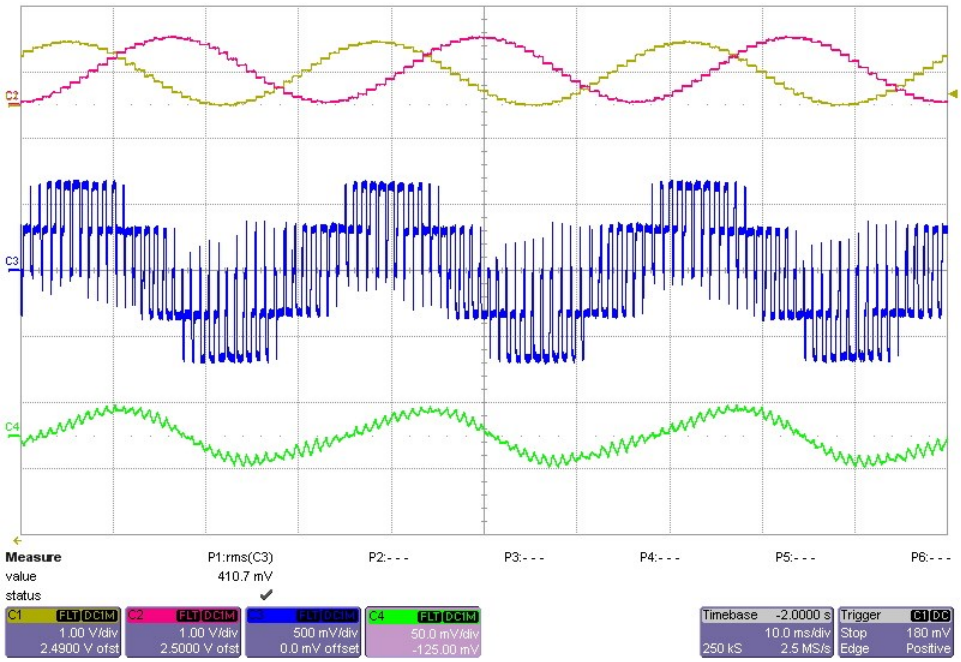
U_{out} [V]	f_{out} [Hz]	f_{sw} [kHz]	m [-]	U_{RMS} [V]	Číslo obrázku
50	10	1	1	41,4	19
30	10	1	0,6	31,8	20
50	30	1	1	41,1	21
30	30	1	0,6	31,2	22



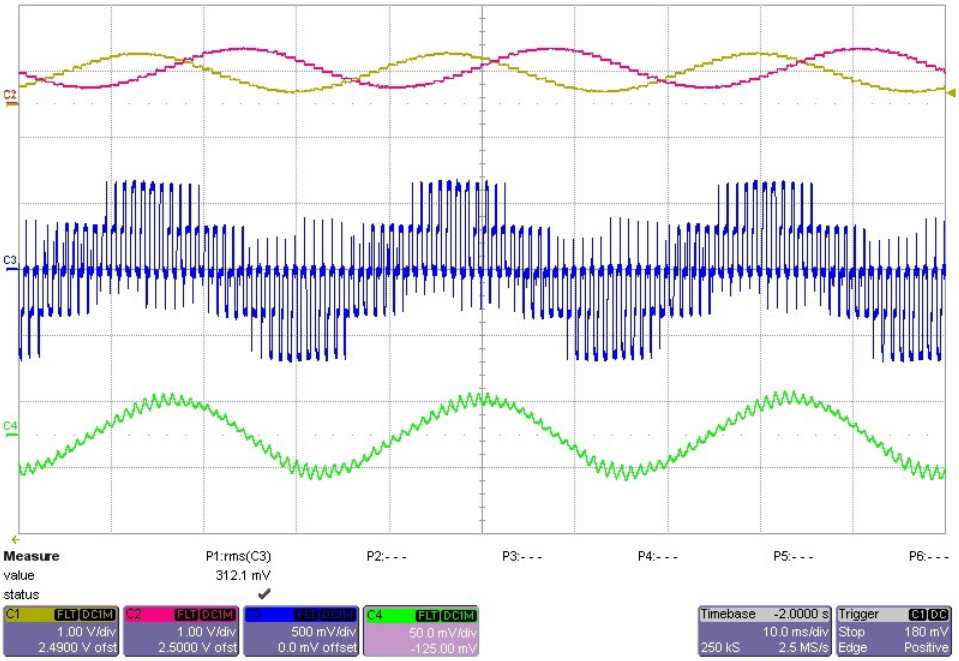
Obr. 19 - Průběhy pro amplitudu výstupního napětí 50 V při výstupní frekvenci 10 Hz a f_{SW} = 1 kHz



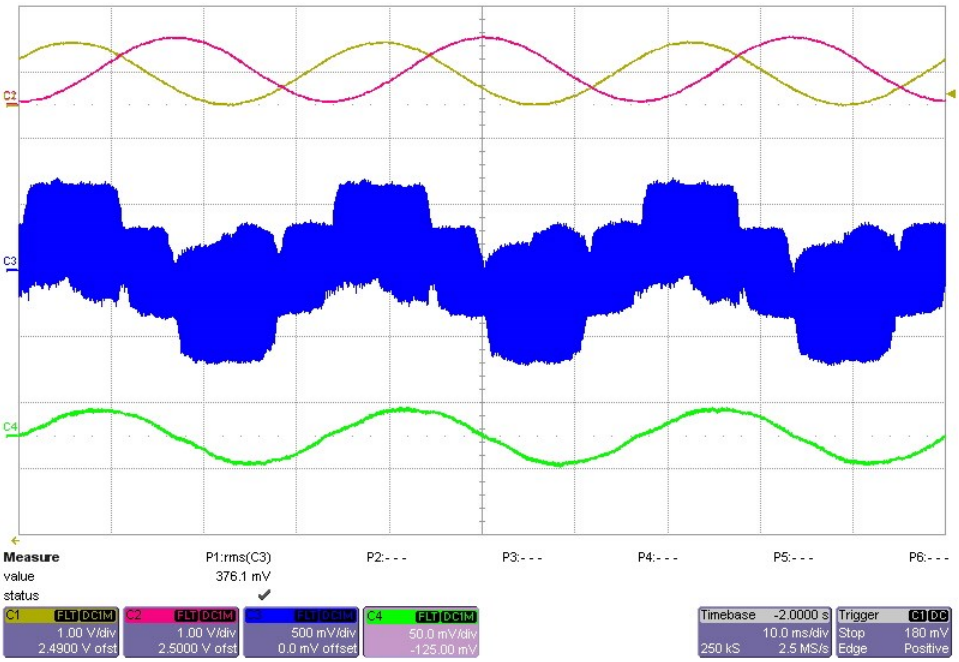
Obr. 20 - Průběhy pro amplitudu výstupního napětí 30 V při výstupní frekvenci 10 Hz a f_{SW} = 1 kHz



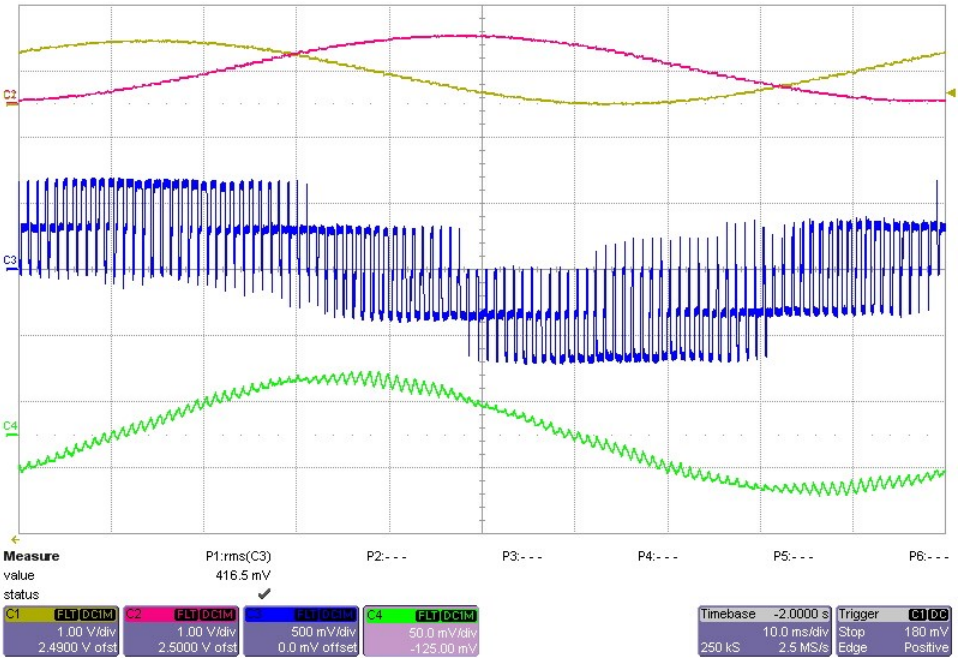
Obr. 21 - Průběhy pro amplitudu výstupního napětí 50 V při výstupní frekvenci 30 Hz a f_{SW} = 1 kHz



Obr. 22 - Průběhy pro amplitudu výstupního napětí 30 V při výstupní frekvenci 30 Hz a f_{SW} = 1 kHz



Obr. 23 - Průběhy pro amplitudu výstupního napětí 50 V při výstupní frekvenci 30 Hz a f_{SW} = 10 kHz



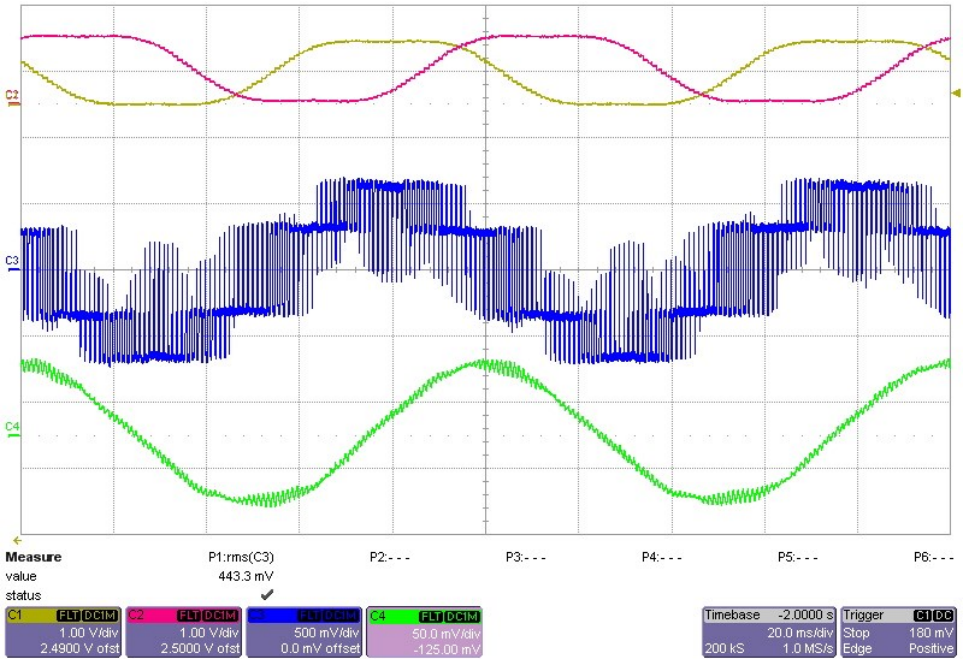
Obr. 24 - Detail modulace pro amplitudu výstupního napětí 50 V při výstupní frekvenci 10 Hz a f_{SW} = 1 kHz

7.2 Komparační PWM s přidáním třetí harmonické

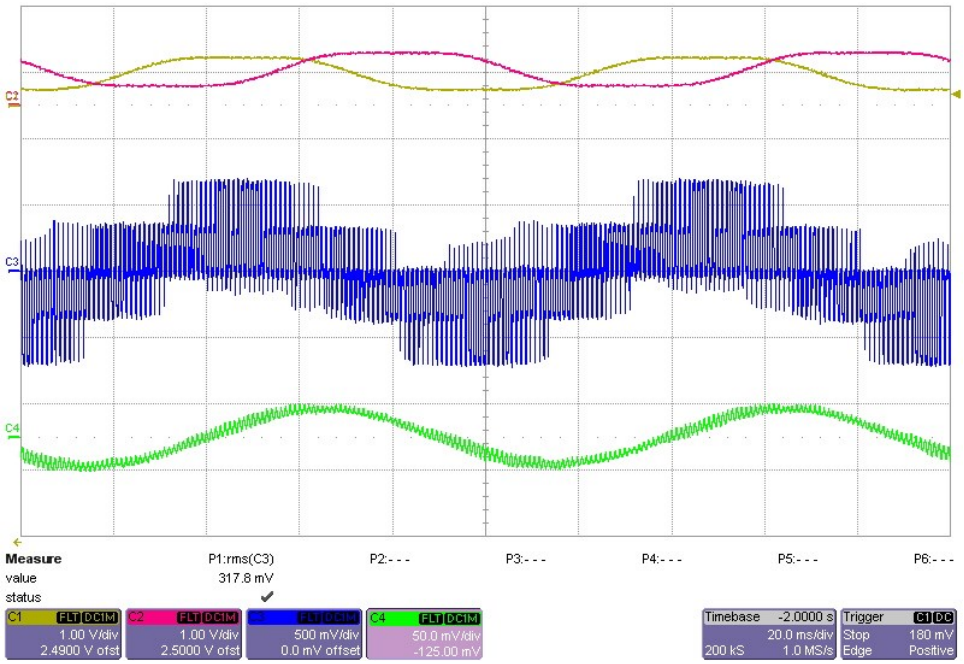
Průběhy pro různé parametry výstupního signálu jsou uvedeny na obrázcích 25 až 28. V tabulce 6 jsou pak shrnuty získané výsledky. Na obrázku 29 je zaznamenán průběh pro f_{sw} = 10 kHz a na obrázku 30 pak detail modulace výstupního napětí.

Tabulka 6 - Shrnutí naměřených výsledků

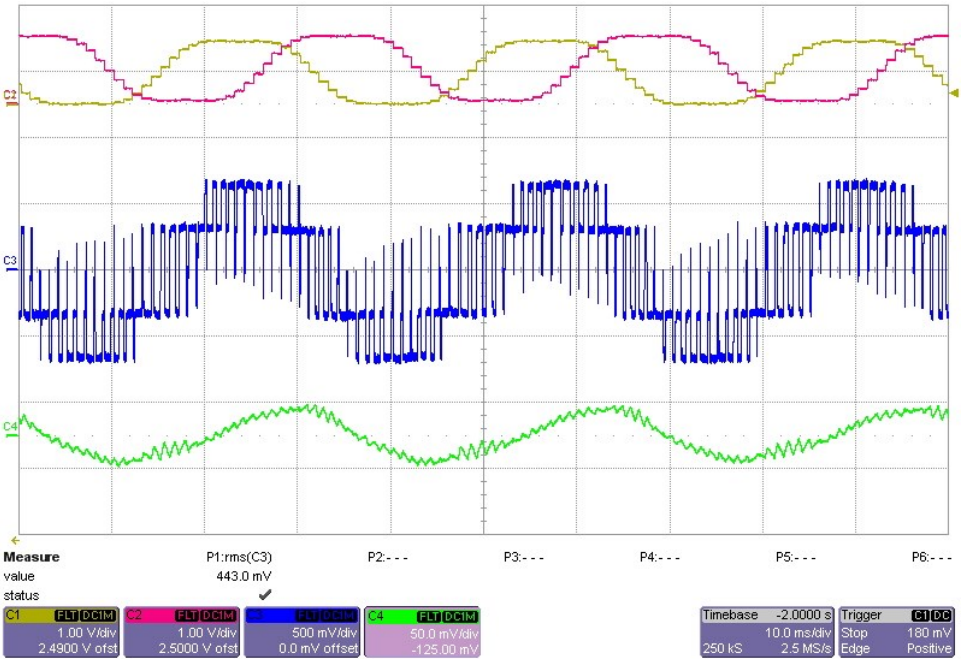
U_{out} [V]	f_{out} [Hz]	f_{sw} [kHz]	m [-]	U_{RMS} [V]	Číslo obrázku
50	10	1	1	44,3	25
30	10	1	0,52	31,8	26
50	30	1	1	44,3	27
30	30	1	0,52	31,1	28



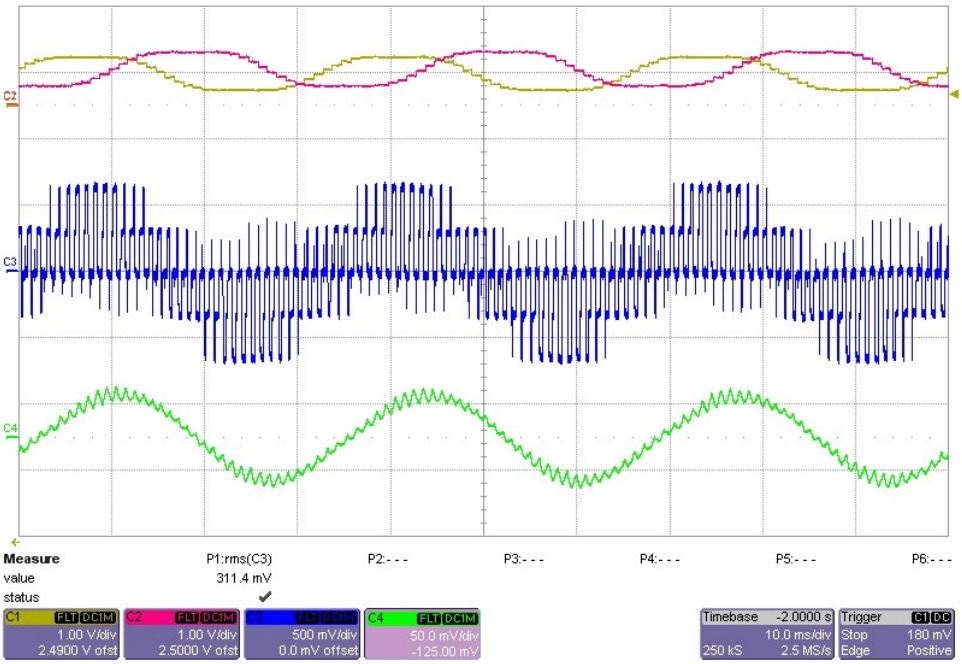
Obr. 25 - Průběhy pro amplitudu výstupního napětí 57,7 V při výstupní frekvenci 10 Hz a f_{SW} = 1 kHz



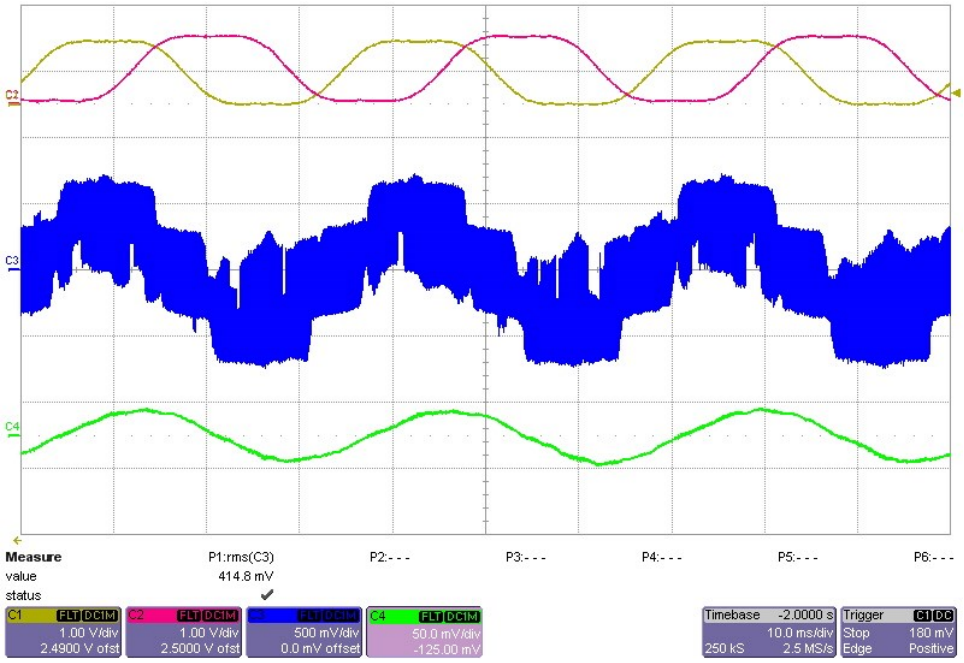
Obr. 26 - Průběhy pro amplitudu výstupního napětí 30 V při výstupní frekvenci 10 Hz a f_{SW} = 1 kHz



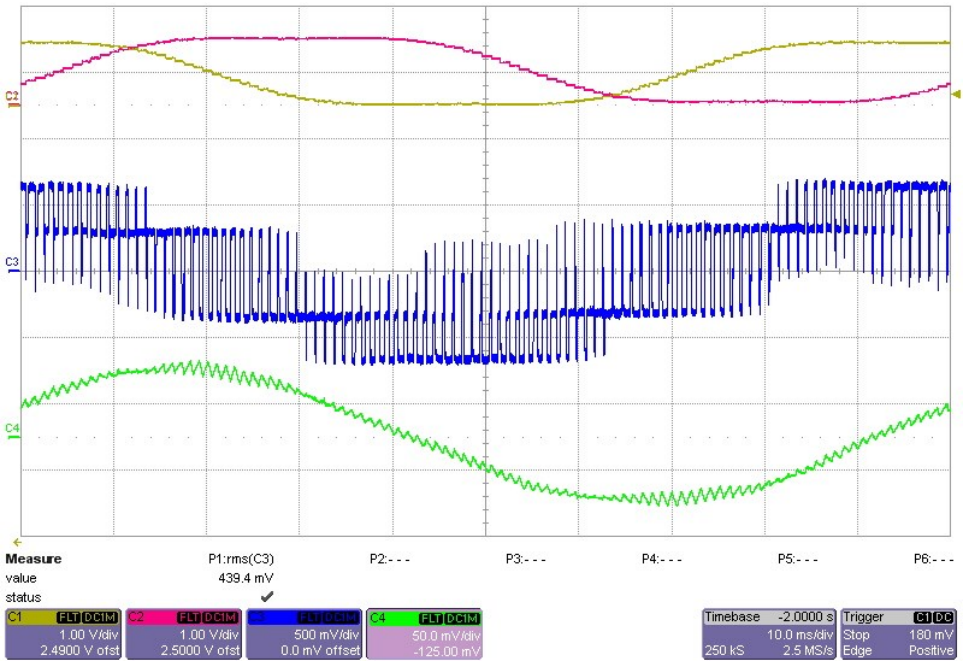
Obr. 27 - Průběhy pro amplitudu výstupního napětí 57,7 V při výstupní frekvenci 30 Hz a f_{SW} = 1 kHz



Obr. 28 - Průběhy pro amplitudu výstupního napětí 30 V při výstupní frekvenci 30 Hz a f_{SW} = 1 kHz



Obr. 29 - Průběhy pro amplitudu výstupního napětí 30 V při výstupní frekvenci 30 Hz a f_{SW} = 10 kHz



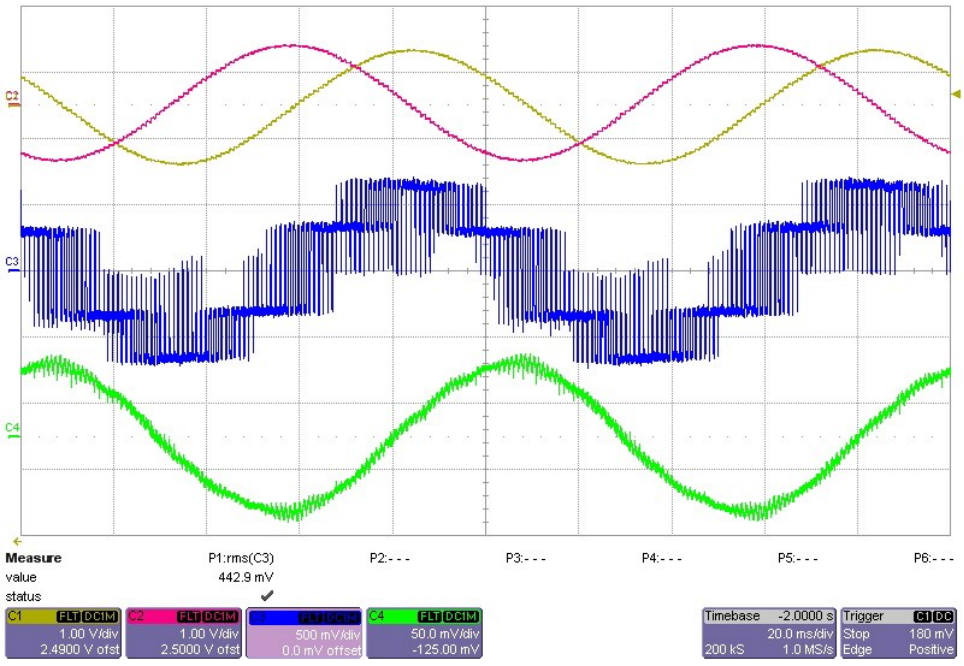
Obr. 30 - Detail modulace pro amplitudu výstupního napětí 57,7 V při výstupní frekvenci 10 Hz a f_{SW} = 1 kHz

7.3 Vektorová PWM s využitím kompetitivní neuronové sítě

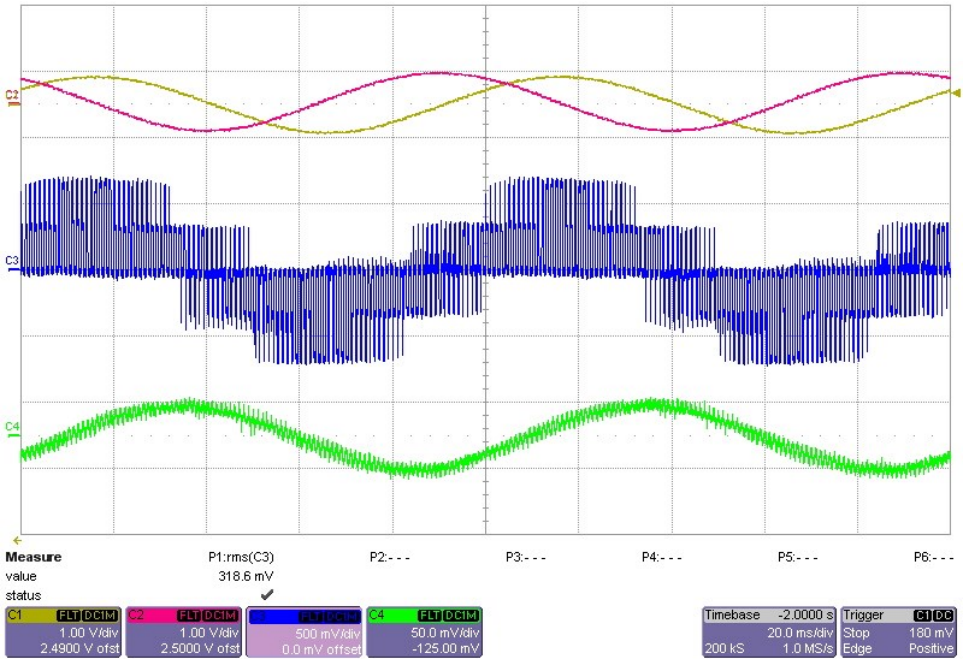
Průběhy pro různé parametry výstupního signálu jsou uvedeny na obrázcích 31 až 34. V tabulce 7 jsou pak shrnuty získané výsledky. Na obrázku 35 je zaznamenán průběh pro f_{sw} = 10 kHz a na obrázku 36 pak detail modulace výstupního napětí.

Tabulka 7 - Shrnutí naměřených výsledků

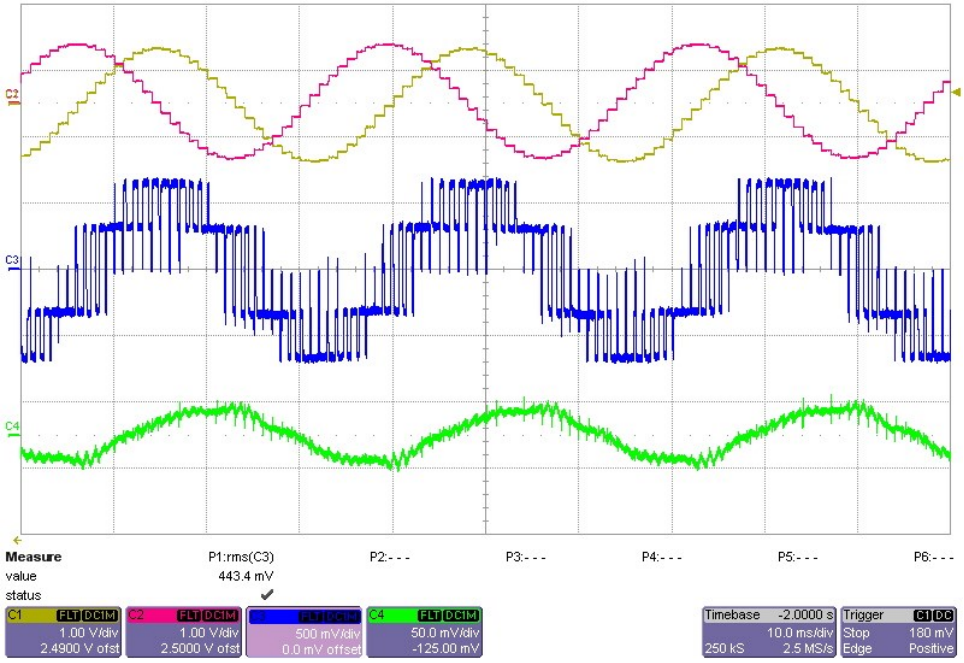
U_{out} [V]	f_{out} [Hz]	f_{sw} [kHz]	m [-]	U_{RMS} [V]	Číslo obrázku
50	10	1	1	44,3	31
30	10	1	0,52	31,8	32
50	30	1	1	44,3	33
30	30	1	0,52	31,2	34



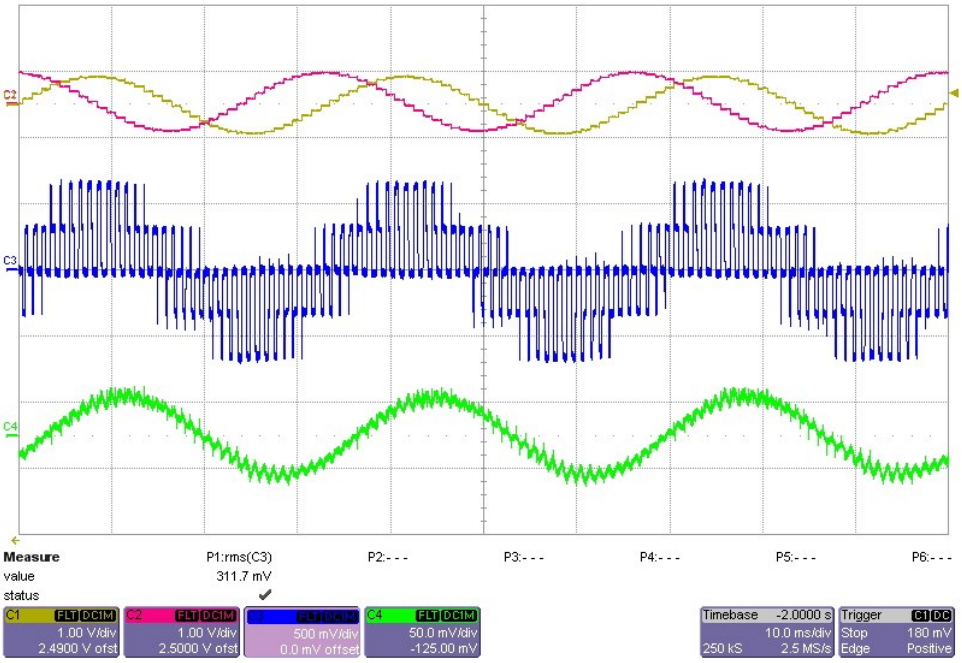
Obr. 31 - Průběhy pro amplitudu výstupního napětí 57,7 V při výstupní frekvenci 10 Hz a f_{SW} = 1 kHz



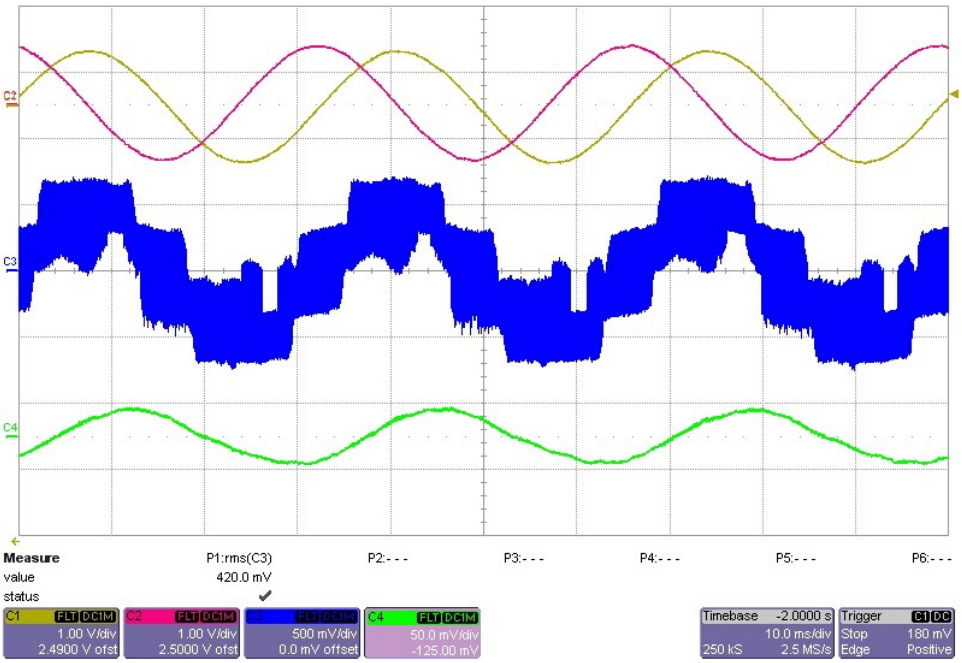
Obr. 32 - Průběhy pro amplitudu výstupního napětí 30 V při výstupní frekvenci 10 Hz a f_{SW} = 1 kHz



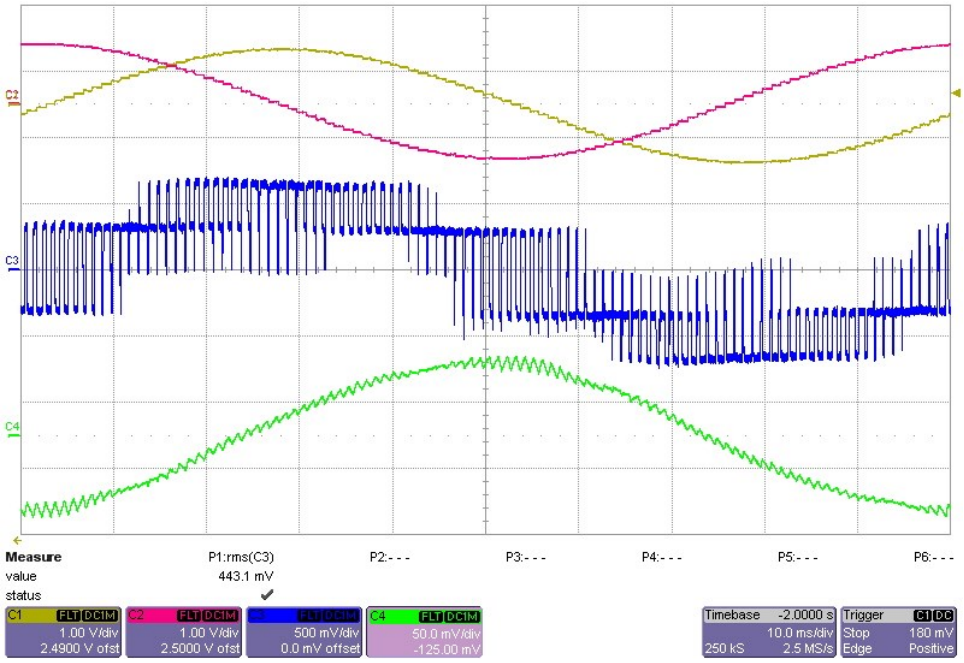
Obr. 33 - Průběhy pro amplitudu výstupního napětí 57,7 V při výstupní frekvenci 30 Hz a f_{SW} = 1 kHz



Obr. 34 - Průběhy pro amplitudu výstupního napětí 30 V při výstupní frekvenci 30 Hz a f_{SW} = 1 kHz



Obr. 35 - Průběhy pro amplitudu výstupního napětí 57,7 V při výstupní frekvenci 30 Hz a f_{SW} = 10 kHz



Obr. 36 - Detail modulace pro amplitudu výstupního napětí 57,7 V při výstupní frekvenci 10 Hz a f_{SW} = 1 kHz

7.4 Zhodnocení získaných výsledků

Z tabulek 6 a 7 je zřejmé, že maximální efektivní hodnota výstupního napětí NMK je pro komparační metodu s třetí harmonickou a metodu vektorové PWM shodná, což odpovídá teoretickému předpokladu.

Po srovnání maximální dosažitelné efektivní hodnoty výstupního napětí NMK při použití komparační metody a při použití zbylých dvou metod vychází, že zlepšení využití napětí meziobvodu je pouze 7% a ne 15% jak je předpokládáno. Tato odchylka od teoretického předpokladu je způsobena nepřesností měření U_{RMS} osciloskopem, který kvůli vysokému počtu pulzů není schopen tuto hodnotu přesně spočítat.

Využití napětí meziobvodu však je skutečně o 15% vyšší, což dokládají výsledky dosažené při nastaveném výstupním napětí 30 V. V tom případě má totiž U_{RMS} u všech metod stejnou velikost, přestože u komparační metody s třetí harmonickou a u vektorové PWM je hloubka modulace o 15% nižší než u komparační PWM.

Z průběhů proudů je zřejmé, že měření probíhá na motoru s nevyvedeným uzlem, protože fázový proud motoru se u žádné z metod, co se tvaru týče, nijak neliší.

Z naměřených průběhů je dále zřejmé, že zvýšení spínací frekvence má za následek zvýšení přesnosti konstruovaných referenčních napětí, protože jejich výpočty probíhají během jedné periody ve více vzorcích. Vliv nízké spínací frekvence na aliasing vypočených referenčních napětí je zřetelný na obrázcích 22, 28 a 33.

8 Závěr

Při tvorbě této diplomové práce si student osvojil znalosti týkající se metod řízení výstupního napětí NMK s napěťovým meziobvodem. Pro úspěšné nastavení digitálního signálového procesoru TMS320F28335 bylo třeba nastudovat a pochopit funkce jednotlivých periferií, které byly následně při implementaci zvolených metod řízení využity. Zvolené metody řízení student následně implementoval do řídicího systému osazeného DSP. Během implementace metod do zmíněného DSP studen využil své znalosti programování v jazyce C. Při tvorbě uživatelského rozhraní pro PC pak byly využity znalosti programování v prostředí Labview, které si student osvojil v předešlých letech svého studia.

Práce je psána v takovém sledu, v jakém student postupoval při řešení zadané problematiky. Nejprve jsou rozebrány způsoby řízení výstupního napětí NMK s napěťovým meziobvodem a následně je popsána funkce periferií nezbytných k realizaci rozebíraných metod. Díky získaným znalostem pak student rozebere provedená nastavení periferií DSP, která byla zvolena jako nejoptimálnější pro pozdější implementaci samotných algoritmů řízení výstupního napětí NMK. V následujících kapitolách je pak popsána realizace jednotlivých metod a obslužného software.

Vytvořené algoritmy byly ověřeny měřením na laboratorním stanovišti s asynchronním motorem a výsledky měření dokládají jak funkčnost realizovaného aplikačního software, tak teoretické předpoklady.

Dalším vylepšením práce by mohla být realizace optimálního spínání napěťových vektorů při komparační PWM nebo doplnění funkce frekvenčního řízení s konstantním poměrem U/f.

Literatura

[1] BRANDŠTETTER Pavel. *Elektrické pohony III,* Vysoká škola báňská - Technická univerzita Ostrava, Ostrava, 2012, 201 stran. [cit 11.4.2016].

[2] Autor neuveden. *Princip - Frekvenční měnič s pulzně šířkovou modulací* [online].Datum neuvedeno, [cit. 12.4.2016]. Dostupné z: < http://www.pohonnatechnika.cz/frekvencni-menice/princip-frekvencniho-menice/princip-pwm-menice >

[3] Kuchař, Martin. Disertační práce: *Řídicí aplikace neuronových sítí v elektrických regulačních pohonech*, VŠB-TU Ostrava, Ostrava, 2003, 101 stran.

[4] Texas Instruments. *TMS320x2833x, 2823x Enhanced Pulse Width Modulator (ePWM) Module Reference Guide* [online]. Říjen 2008, [cit. 12.4.2016]. Dostupné z: < http://www.ti.com/lit/ug/sprug04a/sprug04a.pdf >

[5] Texas Instruments. *TMS320F28335, TMS320F28334, TMS320F28332, TMS320F28235, TMS320F28234, TMS320F28232 Digital Signal Controllers (DSCs) Data Manual* [online]. Červen.2007, [cit. 12.4.2016]. Dostupné z: < http://www.ti.com/lit/ds/symlink/tms320f28335.pdf >

[6] CT-Concept Technlogie AG. *Six-pack SCALE Driver 6SD106EI* [online]. 13.4.2016, [cit. 20.4.2016]. Dostupné z: < https://igbt-driver.power.com/system/files_force/product_document/data_sheet/6SD106EI.pdf >

Seznam příloh

I. Zdrojový kód aplikačního software DSP a Uživatelské rozhraní pro PC

Příloha na CD/DVD